2017-01-09: CBOR WG

- Concise Binary Object Representation Maintenance and Extensions

1. Formal process: Take RFC 7049 to IETF STD level

2. Standardize CDDL as a data definition language

3. (Maybe define a few more CBOR tags, as needed.)

CDDL

Henk Birkholz, Christoph Vigano, Carsten Bormann

draft-ietf-cbor-cddl

ABNF

- BNF (Backus-Naur form) : grammars for strings
  - RFC40 (1970): first RFC with BNF
- “Internet” BNF: Augmented BNF (ABNF)
  - RFC 733 (1977): “Ken L. Harrenstien, of SRI International, was responsible for re-coding the BNF into an augmented BNF which compacts the specification and allows increased comprehensibility.”

ABNF in the IETF

- 752 RFCs and I-Ds reference RFC 5234 (the most recent version of ABNF) [cf. YANG: 160]

- Tool support (e.g., BAP, abnf-gen; antlr support)

- Pretty much standard for text-based protocols that aren’t based on XML or JSON

ABNF is composed of productions

```
addr-spec      = local-part "@" domain
local-part     = dot-atom / quoted-string / obs-local-part
domain         = dot-atom / domain-literal / obs-domain
domain-literal = [CFWS] "[" *([FWS] dtext) [FWS] "]" [CFWS]
dtext          = %d33-90 /          ; Printable US-ASCII
                 %d94-126 /         ;  characters not including
                 obs-dtext          ;  "[", "]", or “\"
```

- **Names** for sublanguages
- **Compose** using
  - Concatenation
  - Choice: /
- **Literals** terminate nesting

From ABNF to CDDL

- Build **trees** of data items, not **strings** of characters
- Add literals for primitive types
- Add constructors for containers (arrays, maps)
  - Inspiration: Relax-NG (ISO/IEC 19757-2)

Rule names are **types**

```
bool = false / true
label = text / int
int = uint / nint
```

- Types are **sets** of potential values
- Even literals are (very small) types

```
participants = 1 / 2 / 3
participants = 1..3
msgtype = "PUT"
msgtype = 1
```

Groups: building containers

- Containers contain sequences (array) or sets (maps) of entries
- Entries are types (array) or key/value type pairs (maps)
- Unify this into **group**:
  - sequenced (ignored within maps)
  - labeled (ignored within arrays)

How **RFC 7071** would have looked like in CDDL

```
reputation-object = {              ; This is a map (JSON object)
  application: text                ;  text string (vs. binary)
  reputons: [* reputon]            ;  Array of 0-∞ reputons
}

reputon = {                        ; Another map (JSON object)
  rater: text
  assertion: text
  rated: text
  rating: float16                  ; OK, float16 is a CBORism
  ? confidence: float16            ; optional…
  ? normal-rating: float16
  ? sample-size: uint              ; unsigned integer
  ? generated: uint
  ? expires: uint
  * text => any                    ; 0-∞, express extensibility
}
```

Named groups

```
header_map = {
    Generic_Headers,
    * label => values
}
Generic_Headers = (
    ? 1 => int / tstr,  ; algorithm identifier
    ? 2 => [+label],    ; criticality
    ? 3 => tstr / int,  ; content type
    ? 4 => bstr,        ; key identifier
    ? 5 => bstr,        ; IV
    ? 6 => bstr,        ; Partial IV
    ? 7 => COSE_Signature / [+COSE_Signature]
)
```

- Named groups allow re-use of parts of a map/array
- Inclusion instead of inheritance

GRASP

- Generic Autonomic Signaling Protocol (GRASP)
- For once, try not to invent another TLV format: just use CBOR
- Messages are arrays, with type, id, option:

```
message /= [MESSAGE_TYPE, session-id, *option]
MESSAGE_TYPE = 123 ; a defined constant
session-id = 0..16777215
; option is one of the options defined below
```

- Options are arrays, again:

```
option /= waiting-time-option
waiting-time-option = [O_WAITING, waiting-time]
O_WAITING = 456 ; a defined constant
waiting-time = 0..4294967295  ; in milliseconds
```

References to draft-greevenbosch-appsawg-cbor-cddl

This is an experimental product. These dependencies are extracted using heuristics looking for strings with particular prefixes. Notably, this means that references to I-Ds by title only are not reflected here. If it's really important, please inspect the documents' refere

Showing RFCs and active Internet-Drafts, sorted by reference type, then document name.

Document	Title	Status	Type
draft-bormann-cbor-time-tag	**Concise Binary Object Representation (CBOR) Tags for Time, Duration, and Period** Refs Ref'd by		normatively references
draft-bormann-lpwan-cbor-template	**Concise Binary Object Representation (CBOR) Tag for CBOR Templates** Refs Ref'd by		normatively references
draft-bormann-t2trg-sworn	**SWORN: Secure Wake on Radio Nudging** Refs Ref'd by		normatively references
draft-carpenter-anima-ani-objectives	**Technical Objective Formats for the Autonomic Network Infrastructure** Refs Ref'd by		normatively references
draft-jroatch-cbor-tags	**Concise Binary Object Representation (CBOR) Tags for Typed Arrays** Refs Ref'd by		normatively references
draft-mandyam-tokbind-attest	**Attested TLS Token Binding** Refs Ref'd by		normatively references
draft-birkholz-tuda	**Time-Based Uni-Directional Attestation** Refs Ref'd by		informatively references
draft-hartke-t2trg-cbor-forms	**CBOR-encoded Form Data** Refs Ref'd by		informatively references
draft-hartke-t2trg-coral	**The Constrained RESTful Application Language (CoRAL)** Refs Ref'd by		informatively references
draft-ietf-ace-cbor-web-token	**CBOR Web Token (CWT)** Refs Ref'd by		informatively references
draft-ietf-anima-grasp	**A Generic Autonomic Signaling Protocol (GRASP)** Refs Ref'd by	Proposed Standard	informatively references
draft-ietf-anima-prefix-management	**Autonomic IPv6 Edge Prefix Management in Large-scale Networks** Refs Ref'd by	Informational	informatively references
draft-ietf-core-links-json	**Representing CoRE Formats in JSON and CBOR** Refs Ref'd by	Proposed Standard	informatively references
draft-ietf-core-object-security	**Object Security of CoAP (OSCOAP)** Refs Ref'd by		informatively references
draft-ietf-core-senml	**Media Types for Sensor Measurement Lists (SenML)** Refs Ref'd by	Proposed Standard	informatively references
draft-ietf-cose-msg	**CBOR Object Signing and Encryption (COSE)** Refs Ref'd by	Proposed Standard	informatively references
draft-ietf-dnsop-dns-capture-format	**C-DNS: A DNS Packet Capture Format** Refs Ref'd by	Proposed Standard	informatively references
draft-ietf-sacm-coswid	**Concise Software Identifiers** Refs Ref'd by		informatively references
draft-liu-opentrustprotocol-cbor	**Open Trust Protocol CBOR Encoding** Refs Ref'd by		informatively references
draft-schaad-cose-x509	**CBOR Encoded Message Syntax (COSE): Headers for carrying and referencing X.509 certificates** Refs Ref'd by		informatively references
draft-selander-ace-cose-ecdhe	**Ephemeral Diffie-Hellman Over COSE (EDHOC)** Refs Ref'd by		informatively references
rfc8007	**Content Delivery Network Interconnection (CDNI) Control Interface / Triggers** Refs Ref'd by	Proposed Standard	informatively references

SDOs outside of IETF

- CDDL is being used for specifying both CBOR and JSON in W3C, ___, and _____ ___
- Data in flight in a variety of protocols, e.g.
  - Access to specific features in wireless radios
  - Aggregation of metadata,
    enabling visualization of network topologies

From draft to RFC

- **Do not**: break it
- Editorial improvements required
- Any additional language features needed?
  - Should stay in the “tree grammar” envelope
  - Should be mostly done with that, anyway.
- What can we take out?
  Not much without breaking specs.

Avoid the kitchen sink

- This is not a Christmas wish list
- Each feature has a cost
  - specification complexity
  - learning effort
  - implementation effort

Improvements of definition

- https://cbor-wg.github.io/cddl/matching/draft-ietf-cbor-cddl.html#rfc.appendix.B
- Editors' draft, "matching" branch: new appendix B, matching rules
  - Concisely summarizes CDDL semantics
- Is this
  - Useful
  - Correct
  - Complete?

“Map validation” issue

- CDDL semantics are generative (production system)
- All elements of a group in a map are equal
- Wildcard match (for extensibility) can enable what was not intended to be enabled

```
{ ? 4=>text,
  * uint=>any }
```

- How to create priority for “more specific”?

cuts (better error messages)

```
a = ant / cat / elk
ant = ["ant", ^ uint]
cat = ["cat", ^ text]
ant = ["elk", ^ float]
```

```
["ant", 47.11]
```

- Tool will not tell you "can't match a",
  but "can't match rest of ant”

- Worth adding?

Proposal: use cuts here, too

- A cut after recognizing a map key cuts off any alternative matches

```
{ ? 4 ^ =>text,
  * uint=>any }
```

- Proposal: Make existing ":" a shortcut for "^ =>"

```
{ ? 4: text,
  * uint=>any }
```

- TO DO: fully define
- TO DO: check for breakage
- TO DO: implement

CBOR (RFC 7049) bis

Concise Binary Object Representation

Carsten Bormann, 2017-11-16

Take CBOR to STD

- **Do not**: futz around
- **Do**:
  - Document interoperability
  - Make needed improvements in specification quality
    - At least fix the errata :-)
  - Check: Are all tags implemented interoperably?

Take CBOR to STD

Process as defined by RFC 6410**:**

- independent interoperable implementations ✔
- no errata (oops) ✔ in draft
- no unused features [_]
- (if patented: licensing process) [N/A]

draft-ietf-cbor-7049bis-01

- –00 had already fixed errata
- –01: 2017-10-14
- Amplification of chosen Simple encoding (1-byte only for false/true/null etc.)
- Add a changes section
    - Maybe sort this into fixes and new information?
- New: Section 2.5 CBOR Data Models

CBOR data models

- Biggest failing of JSON: Data model now entirely implicit
- Observant reader could infer CBOR data model from RFC 7049
- Now more explicit: “generic data model” (as opposed to any specific data model realized in CBOR)
- Unextended (basic) data model
- Extension points: Simple, Tags
  - Pre-extension by false/true/null/undefined, 18 pre-defined tags
  - Further extension by Simple/Tag definitions (IANA)

Why is a generic data model important?

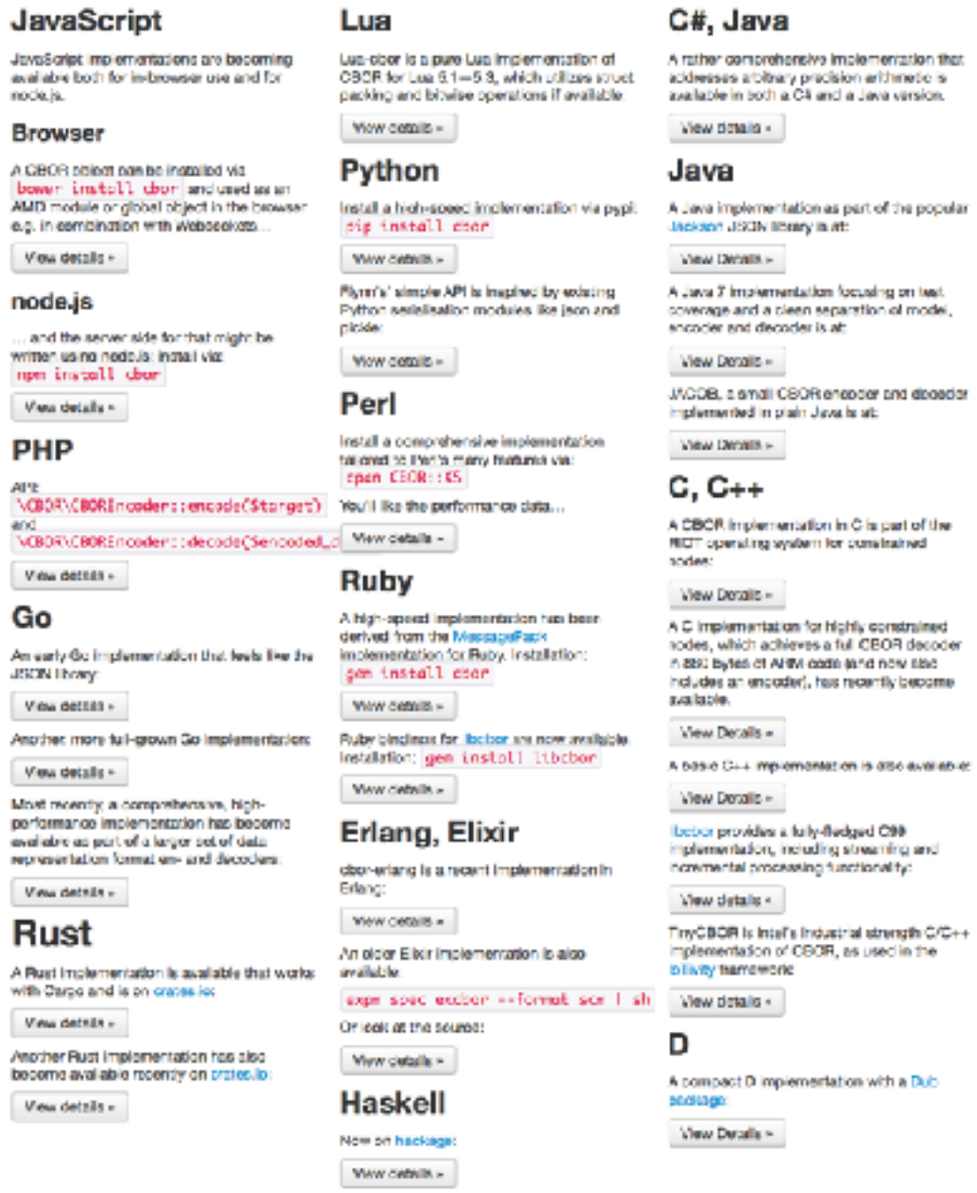
- Generic data model enables the implementation of generic encoders and decoders
- An ecosystem of generic encoders and decoders
  - makes interoperability so much more likely
  - guides definition of specific data models

“Expectations”

- “Batteries included”: not always appropriate
- But some of the pre-extensions are really basic
  - Which ones?
- Section 2.5 states **false/true/null** are *expected* to be provided in a generic encoder/decoder
- Anything else (Simple: **undefined**, 18 tags) is “truly optional and a matter of implementation quality”.

Implementations

- Parsing/generating CBOR easier than interfacing with application
  - Minimal implementation: 822 bytes of ARM code
- Different integration models, different languages
- > 40 implementations



http://cbor.io

Houston, we have an interoperability problem

- Tags 21, 22, 33, 34: base64url, base64 classic
- Those can be used with or without padding. Which one is it?
- Defined for tag 21: base64url **without** padding.
- But what about tag 22, 34? Reference to RFC4648 not helpful.
- Tag 33: is this also limited to base64url without padding? (And what about tag 34?)
- (Is white space allowed? I don't think so. Weird line length limitations? Of course not.)

Being permissive is not solving this

- Tag 21, 22 are intended to be acted upon by a CBOR-to-JSON converter — need to know how
- Tag 33, 34 could be interpreted in a more permissive way?
    - Depending on specific data model, might require re-encoding on conversion to JSON (!)

How are base64, base64url being used in practice?

- Easy: Base64url is almost always without padding
  - Interoperability benefits from nailing this down
- Base64 more variable
  - Usually used with padding, but exceptions
  - Bikeshed

Solutions?

- Be more explicit about tag 33: base64url is used without padding in this case, too

- Could define tag 22/34 as with or without padding
    - Tag 22 defines JSON side, tag 34, CBOR side

- Could define additional tags for padding/none (probably only for base64 classic)

- Also, tag 23 (base16): lower or upper case?

Proposal

- Padding designed to help with indeterminate length
- We do know the length, so *no padding* is “right”
- RFC 7049 was unclear about this
- ➔ for base64 classic, go for no padding, too
  - add an implementation note explaining the clarification and asking to be particularly liberal about what you accept

Continuing work on implementation matrix

- https://github.com/cbor-wg/CBORbis/wiki/Implementation-matrix
- Need to fill in more columns
  - Certainly not for all 45 implementations :-)
- Who?

cbor-wg / CBORbis

Unwatch 6

Code Issues 1 Pull requests 1 Projects 0 Wiki Settings

Implementation matrix

Joe Hildebrand edited this page on Jul 18 · 3 revisions

D = Decode E = Encode

Feature	TinyCBOR	node-cbor	cbor-ruby	impl4
Major type 0 (uint)		DE		
Major type 1 (nint)		DE		
Major type 2 (bstr)		DE		
Major type 3 (tstr)		DE		
Major type 4 (array)		DE		
Major type 5 (map)		DE		
Major type 6 (tag)		DE		
Major type 7 (simple)		DE		
Float16		DE		
Float32		DE		
Float64		DE		
Indefinite length array/map		D		
Indefinite length string		D		
Canonical CBOR		DE		
Tag 0		D		
Tag 1		DE		
Tag 2		DE		
Tag 3		DE		
Tag 4		DE		
Tag 5		D		
Tag 21				
Tag 22				
Tag 23				
Tag 24				
Tag 32		DE		
Tag 33				
Tag 34				
Tag 35		DE		

CBOR tag definitions

Carsten Bormann, 2017-11-16

Batteries included

- RFC 7049 predefines 18 Tags
  - Time, big numbers (bigint, float, decimal), various converter helpers, URI, MIME message
- Easy to register your own CBOR Tags
  - > 20 more tags: 6 for COSE; UUIDs, Sets, binary MIME, Perl support, language tagged string, compression

CWT: CBOR Web Token

- JWT: JSON Web Token (RFC 7519)
  - Package **Claim Set** into JSON
  - Apply JOSE for Signing and Encryption
- CWT: Use CBOR and COSE instead of JSON and JOSE
- CWT can replace unstructured misuse of certificates for Claim Sets
- Tag 61 assigned; WGLC completed in IETF ACE WG (draft-ietf-ace-cbor-web-token)

Status of Tags drafts

- OID: On charter, kitchen sink, expired.
  Needs work.

- **Array**: On charter, ready for adoption

- **Time**: Off charter; solved for now by FCFS registration (3-byte tag 1001); move spec to RFC how?

- Template: Off charter
  (will likely be done with SCHC anyway)

- "**Useful tags**": Maybe document some of the more useful registered tags in an RFC on its own (could include Time)?

draft-jroatch-cbor-tags-06

- Provide tags for homogeneous arrays represented in byte strings
  - Inspired by JavaScript
  - 12×2: Both LSB and MSB first
  - Reserves 24 contiguous tags in 2-byte space

uint	sint	float
uint8	sint8	binary16
uint16	sint16	binary32
uint32	sint32	binary64
uint64	sint64	binary128

- Provides a tag for other homogeneous arrays
- Provides a tag for multidimensional arrays

Array tags: 2-byte space?

- 2-byte Tags: Tags 24 to 255
- 2017: ~ 20 taken of 232; be careful with the space
- This is taking out 24 more — would this be a waste of 2-byte space?
  - **Yes**; arrays can be large; fine with 3-byte tags
  - **No**; arrays can also be small (e.g., RGB)
- Could partition 2 vs. 3 by size of basic type; ugly
- Would like to move this ahead (technical decision should not be an obstacle for draft adoption anyway)

Time tag

- Document 1001 as is
  - Could do this on independent stream, WG allowing
- Develop 1001 into a more general time tag

http://cbor.io

http://cbor.me

http://cddl.space