Network Working Group
Internet-Draft
Intended status: Informational
Expires: 27 January 2021

C. Bormann
Universität Bremen TZI

26 July 2020

Packed CBOR
draft-bormann-cbor-packed-01

Abstract

The Concise Binary Object Representation (CBOR, RFC 7049) 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 does not provide any forms of data compression. CBOR data items, in particular when generated from legacy data models often allow considerable gains in compactness when applying data compression. While traditional data compression techniques such as DEFLATE (RFC 1951) work well for CBOR, their disadvantage is that the receiver needs to unpack the compressed form to make use of data.

This specification describes Packed CBOR, a simple transformation of a CBOR data item into another CBOR data item that is almost as easy to consume as the original CBOR data item. A separate decompression step is therefore often not required at the receiver.

Note to Readers

This is an individual submission to the CBOR working group of the IETF, https://datatracker.ietf.org/wg/cbor/about/ (https://datatracker.ietf.org/wg/cbor/about/). Discussion currently takes places on the github repository https://github.com/cabo/cbor-packed (https://github.com/cabo/cbor-packed). If the CBOR WG believes this is a useful document, discussion is likely to move to the CBOR WG mailing list and a github repository at the CBOR WG github organization, https://github.com/cbor-wg (https://github.com/cbor-wg).

The current version is true work in progress; some of the sections haven't been filled in yet, and in particular, permission has not been obtained from tag definition authors to copy over their text.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

(TO DO, expand on text from abstract here; move references here and neuter them in the abstract as per Section 4.3 of [RFC7322].)

The specification defines a transformation from a Packed CBOR data item to the original CBOR data item; it does not define an algorithm for an actual packer. Different packers can differ in the amount of effort they invest in arriving at a minimal packed form.

Packed CBOR can employ two kinds of optimization:

* structure sharing: substructures (data items) that occur repeatedly in the original CBOR data item can be collapsed to a simple reference to a common representation of that data item. The processing required during consumption is limited to following that reference.

* prefix sharing: strings that share a prefix can be replaced by a reference to a common prefix plus the rest of the string. The processing required during consumption is similar to following the prefix reference plus that for an indefinite-length string.

A specific application protocol that employs Packed CBOR might allow both kinds of optimization or limit the representation to structure sharing only.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

The definitions of [I-D.ietf-cbor-7049bis] apply. The term "byte" is used in its now customary sense as a synonym for "octet". Where bit arithmetic is explained, this document uses the notation familiar from the programming language C (including C++14's 0bnnn binary literals), except that, in the plain text form of this document. the operator "^" stands for exponentiation.

2. Packed CBOR

Packed CBOR is defined in CDDL [RFC8610] as in Figure 1:

```
Packed-CBOR = #6.6([rump, [*prefix], *shared])
rump = any
prefix = any
shared = any
```

Figure 1: Packed CBOR in CDDL

(This assumes the allocation of tag number 6, which is motivated further below. Note that the semantics of Tag 6 depend on its content: An integer turns the tag into a shared reference, a string into a prefix reference, and an array into a complete Packed CBOR data item as described above.)

The original CBOR data item can be reconstructed by recursively replacing shared and prefix references encountered in the rump by their defined values.

2.1. Referencing Shared Items

Shared items are stored in the third to last element of the array used as tag content for tag number 6, numbered starting by 2.

The shared data items are referenced by using the data items in Table 1. When reconstructing the original data item, such a reference is replaced by the referenced data item, which is then recursively unpacked.

reference	element number
Simple value 0-15	2-17
Tag 6(unsigned integer N)	18 + 2*N
Tag 6(negative integer N)	18 - 2*N - 1

Table 1: Referencing Shared Values

Taking into account the encoding, there are 16 one-byte references, 48 two-byte references, 512 three-byte references, 131072 four-byte references, etc. As integers can grow to very large (or small) values, there is no practical limit to how many shared items might be used in a Packed CBOR item.

2.2. Referencing Prefix Items

Shared items are stored in an array that is the second element of the array used as tag content for tag number 6. This array is indexed from 0.

Prefix data items are referenced by using the data items in Table 2. When reconstructing the original data item, such a reference is replaced by a string constructed from the referenced prefix data item (prefix, which might need to be recursively unpacked first)

concatenated with the tag content (suffix, again possibly recursively unpacked). The result gets the type of the suffix; this way a single prefix can be used to build both byte and text strings, depending on what type of suffix is being used.

reference	element number
Tag 6(suffix)	0
Tag 224-255(suffix)	1-32
Tag 28672-32767(suffix)	33-4128
Tag 1879048192-2147483647(suffix)	4129-268439584

Table 2: Referencing Prefix Values

Taking into account the encoding, there is one one-byte prefix reference, 32 two-byte references, 4096 three-byte references, and 268435456 five-byte references. 268439585

(2^(28)+2^(12)+2^(5)+2^(0)) is an artificial limit, but should be high enough that there, again, is no practical limit to how many prefix items might be used in a Packed CBOR item.

3. Discussion

This specification uses up a large number of Simple Values and Tags, in particular one of the rare one-byte tags and half of the one-byte simple values. Since the objective is compression, this is warranted if and only if there is consensus that this specific format could be useful for a wide area of applications, while maintaining reasonable simplicity in particular at the side of the consumer.

A maliciously crafted Packed CBOR data item might contain a reference loop. A consumer/decompressor MUST protect against that.

The current definition does nothing to help with packing CBOR sequences [RFC8742]; maybe it should.

Nesting packed CBOR data items is not useful; maybe it should.

4. IANA Considerations

In the registry [IANA.cbor-tags], IANA is requested to allocate the tags defined in Table 3.

Tag	Data Item	Semantics	Reference
6	array, integer, text string, byte string	Packed CBOR: packed/shared/ prefix	draft-bormann-cbor-packed
224-255	text string or byte string	Packed CBOR: prefix	draft-bormann-cbor-packed

28672-32767	text string or byte string	Packed CBOR: prefix	draft-bormann-cbor-packed
1879048192-2147483647	text string or byte string	Packed CBOR: prefix	draft-bormann-cbor-packed

Table 3: Values for Tag Numbers

In the registry [IANA.cbor-simple-values], IANA is requested to allocate the simple values defined in Table 4.

Value	Semantics	Reference
0-15	Packed CBOR: shared	draft-bormann-cbor-packed

Table 4: Simple Values

5. Security Considerations

The security considerations of RFC 7049 apply.

Loops in the Packed CBOR can be used as a denial of service attack, see Section 3.

As the unpacking is deterministic, packed forms can be used as signing inputs. (Note that if external dictionaries are added to cbor-packed, this requires additional consideration.)

6. References

6.1. Normative References

[I-D.ietf-cbor-7049bis]
Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", Work in Progress, Internet-Draft, draft-ietf-cbor-7049bis-14, 16 June 2020, <http://www.ietf.org/internet-drafts/draft-ietf-cbor-7049bis-14.txt>.

[IANA.cbor-simple-values]
IANA, "Concise Binary Object Representation (CBOR) Simple Values", <http://www.iana.org/assignments/cbor-simple-values>.

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

[RFC2119] 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>.

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

[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/info/rfc8610>.

6.2. Informative References

[RFC7322] Flanagan, H. and S. Ginoza, "RFC Style Guide", RFC 7322, DOI 10.17487/RFC7322, September 2014, <https://www.rfc-editor.org/info/rfc7322>.

[RFC8742] Bormann, C., "Concise Binary Object Representation (CBOR) Sequences", RFC 8742, DOI 10.17487/RFC8742, February 2020, <https://www.rfc-editor.org/info/rfc8742>.

Appendix A. Example

The (JSON-compatible) CBOR data structure depicted in Figure 2, 400 bytes of binary CBOR, could lead to a packed CBOR data item depicted in Figure 3, 307 bytes. Note that this example does not lend itself to prefix compression.

```
{ "store": {
    "book": [
      { "category": "reference",
        "author": "Nigel Rees",
        "title": "Sayings of the Century",
        "price": 8.95
      },
      { "category": "fiction",
        "author": "Evelyn Waugh",
        "title": "Sword of Honour",
        "price": 12.99
      },
      { "category": "fiction",
        "author": "Herman Melville",
        "title": "Moby Dick",
        "isbn": "0-553-21311-3",
        "price": 8.99
      },
      { "category": "fiction",
        "author": "J. R. R. Tolkien",
        "title": "The Lord of the Rings",
        "isbn": "0-395-19395-8",
        "price": 22.99
      }
    ],
    "bicycle": {
      "color": "red",
      "price": 19.95
    }
  }
}
```

Figure 2: Example original CBOR data item

```
   6([{"store": {
         "book": [
           {simple(1): "reference", simple(2): "Nigel Rees",
            simple(3): "Sayings of the Century", simple(0): simple(5)},
           {simple(1): simple(4), simple(2): "Evelyn Waugh",
            simple(3): "Sword of Honour", simple(0): 12.99},
           {simple(1): simple(4), simple(2): "Herman Melville",
            simple(3): "Moby Dick", simple(6): "0-553-21311-3",
            simple(0): simple(5)},
           {simple(1): simple(4), simple(2): "J. R. R. Tolkien",
            simple(3): "The Lord of the Rings",
            simple(6): "0-395-19395-8", simple(0): 22.99}],
         "bicycle": {"color": "red", simple(0): 19.95}}},
      [],
      "price", "category", "author", "title", "fiction", 8.95, "isbn"])
      /  0          1         2         3         4       5      6   /
```

Figure 3: Example packed CBOR data item

TBD: Do this for a W3C Thing Description again to get better packing and to exercise prefix compression...

Acknowledgements

CBOR packing was originally invented with the rest of CBOR, but did not make it into [RFC7049]. Various attempts to come up with a specification over the years didn't proceed. In 2017, Sebastian Käbisch proposed investigating compact representations of W3C Thing Descriptions, which prompted the author to come up with essentially the present design.

Author's Address

Carsten Bormann
Universität Bremen TZI
Postfach 330440
D-28359 Bremen
Germany

Phone: +49-421-218-63921
Email: cabo@tzi.org