ï»¿

Internet Engineering Task Force (IETF)
Request for Comments: 9090
Category: Standards Track
ISSN: 2070-1721

C. Bormann
UniversitÃ¤t Bremen TZI

July 2021

Concise Binary Object Representation (CBOR) Tags for Object Identifiers

Abstract

The Concise Binary Object Representation (CBOR), defined in 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.

This document defines CBOR tags for object identifiers (OIDs) and is the reference document for the IANA registration of the CBOR tags so defined.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9090.

Table of Contents

1. Introduction

The Concise Binary Object Representation (CBOR) [RFC8949] provides for the interchange of structured data without a requirement for a pre-agreed schema. [RFC8949] defines a basic set of data types, as well as a tagging mechanism that enables extending the set of data types supported via an IANA registry.

This document defines CBOR tags for object identifiers (OIDs) [X.660], which many IETF protocols carry. The ASN.1 Basic Encoding Rules (BER) [X.690] specify binary encodings of both (absolute) object identifiers and relative object identifiers. The contents of these encodings (the "value" part of BER's type-length-value structure) can be carried in a CBOR byte string. This document defines two CBOR tags that cover the two kinds of ASN.1 object identifiers encoded in this way and a third one to enable a common optimization. The tags can also be applied to arrays and maps to efficiently tag all elements of an array or all keys of a map. This document is the reference document for the IANA registration of the tags so defined.

1.1. 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The terminology of [RFC8949] applies; in particular, the term "byte" is used in its now-customary sense as a synonym for "octet". The verb "to tag (something)" is used to express the construction of a CBOR tag, with the object (something) as the tag content and a tag number indicated elsewhere in the sentence (for instance, in a "with" clause or by the shorthand "an NNN tag" for "a tag with tag number NNN"). The term "SDNV" (Self-Delimiting Numeric Value) is used as defined in [RFC6256], with the additional restriction detailed in Section 2.1 (no leading zeros).

2. Object Identifiers

The International Object Identifier tree [X.660] is a hierarchically managed space of identifiers, each of which is uniquely represented as a sequence of unsigned integer values [X.680]. (These integer values are called "primary integer values" in [X.660] because they can be accompanied by (not necessarily unambiguous) secondary identifiers. We ignore the latter and simply use the term "integer values" here, occasionally calling out their unsignedness. We also use the term "arc" when the focus is on the edge of the tree labeled by such an integer value, as well as in the sense of a "long arc", i.e., a (sub)sequence of such integer values.)

While these sequences can easily be represented in CBOR arrays of unsigned integers, a more compact representation can often be achieved by adopting the widely used representation of object identifiers defined in BER; this representation may also be more amenable to processing by other software that makes use of object identifiers.

BER represents the sequence of unsigned integers by concatenating self-delimiting representations [RFC6256] of each of the integer values in sequence.

ASN.1 distinguishes absolute object identifiers (ASN.1 type "OBJECT IDENTIFIER"), which begin at a root arc ([X.660], Clause 3.5.21), from relative object identifiers (ASN.1 type "RELATIVE-OID"), which begin relative to some object identifier known from context ([X.680], Clause 3.8.63). As a special optimization, BER combines the first two integers in an absolute object identifier into one numeric identifier by making use of the property of the hierarchy that the first arc has only three integer values (0, 1, and 2) and the second arcs under 0 and 1 are limited to the integer values between 0 and 39. (The root arc "joint-iso-itu-t(2)" has no such limitations on its second arc.) If X and Y are the first two integer values, the single integer value actually encoded is computed as:

```
X * 40 + Y
```

The inverse transformation (again making use of the known ranges of X and Y) is applied when decoding the object identifier.

Since the semantics of absolute and relative object identifiers differ and since it is very common for companies to use self-assigned numbers under the arc "1.3.6.1.4.1" (IANA Private Enterprise Number OID [IANA.enterprise-numbers]) that adds 5 fixed bytes to an encoded OID value, this specification defines three tags, collectively called the "OID tags" here:

Tag number 111: Used to tag a byte string as the BER encoding [X.690] of an absolute object identifier (simply "object identifier" or "OID").

Tag number 110: Used to tag a byte string as the BER encoding [X.690] of a relative object identifier (also called "relative OID"). Since the encoding of each number is the same as for Self-Delimiting Numeric Values (SDNVs) [RFC6256], this tag can also be used for tagging a byte string that contains a sequence of zero or more SDNVs (or a more application-specific tag can be created for such an application).

Tag number 112: Structurally like tag 110 but understood to be relative to "1.3.6.1.4.1" (IANA Private Enterprise Number OID [IANA.enterprise-numbers]). Hence, the semantics of the result are that of an absolute object identifier.

2.1. Requirements on the Byte String Being Tagged

To form a valid tag, a byte string tagged with 111, 110, or 112 MUST be syntactically valid contents (the value part) for a BER representation of an object identifier (see Table 1):

Tag number	Section of [X.690]
111	8.19
110	8.20
112	8.20

Table 1: Tag Number and Section of X.690 Governing Tag Content

This is a concatenation of zero or more SDNV values, where each SDNV value is a sequence of one or more bytes that all have their most significant bit set, except for the last byte, where it is unset. Also, the first byte of each SDNV cannot be a leading zero in SDNV's base-128 arithmetic, so it cannot take the value 0x80 (bullet (c) in Section 8.1.2.4.2 of [X.690]).

In other words:

* The byte string's first byte, and any byte that follows a byte that has the most significant bit unset, MUST NOT be 0x80 (this requirement requires expressing the integer values in their shortest form, with no leading zeroes).

* The byte string's last byte MUST NOT have the most significant bit set (this requirement excludes an incomplete final integer value).

If either of these invalid conditions are encountered, the tag is invalid.

[X.680] restricts RELATIVE-OID values to having at least one arc, i.e., their encoding would have at least one SDNV. This specification permits empty relative object identifiers; they may still be excluded by application semantics.

To facilitate the search for specific object ID values, it is RECOMMENDED that definite length encoding (see Section 3.2.3 of [RFC8949]) be used for the byte strings that are used as tag content for these tags.

The valid set of byte strings can also be expressed using regular expressions on bytes, using no specific notation but resembling Perl Compatible Regular Expressions [PCRE]. Unlike typical regular expressions that operate on character sequences, the following regular expressions take bytes as their domain, so they can be applied directly to CBOR byte strings.

For byte strings with tag 111:

```
"/^(([\x81-\xFF][\x80-\xFF]*)?[\x00-\x7F])+$/"
```

For byte strings with tags 110 or 112:

```
"/^(([\x81-\xFF][\x80-\xFF]*)?[\x00-\x7F])*$/"
```

A tag with tagged content that does not conform to the applicable regular expression is invalid.

2.2. Preferred Serialization Considerations

For an absolute OID with a prefix of "1.3.6.1.4.1", representations with both the 111 and 112 tags are applicable, where the representation with 112 will be five bytes shorter (by leaving out the prefix h'2b06010401' from the enclosed byte string). This specification makes that shorter representation the preferred serialization (see Sections 3.4 and 4.1 of [RFC8949]). Note that this also implies that the Core Deterministic Encoding Requirements (Section 4.2.1 of [RFC8949]) require the use of 112 tags instead of 111 tags wherever that is possible.

2.3. Discussion

Staying close to the way object identifiers are encoded in ASN.1 BER makes back-and-forth translation easy; otherwise, we would choose a more efficient encoding. Object identifiers in IETF protocols are serialized in dotted decimal form or BER form, so there is an advantage in not inventing a third form. Also, expectations of the cost of encoding object identifiers are based on BER; using a different encoding might not be aligned with these expectations. If additional information about an OID is desired, lookup services such as the OID Resolution Service (ORS) [X.672] and the OID Repository [OID-INFO] are available.

3. Basic Examples

This section gives simple examples of an absolute and a relative object identifier, represented via tag numbers 111 and 110, respectively.

3.1. Encoding of the SHA-256 OID

```
ASN.1 Value Notation:
   { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
     csor(3) nistalgorithm(4) hashalgs(2) sha256(1) }

Dotted Decimal Notation:  2.16.840.1.101.3.4.2.1

06                                  # UNIVERSAL TAG 6
   09                               # 9 bytes, primitive
      60 86 48 01 65 03 04 02 01    # X.690 Clause 8.19
#      |   840  1  |  3  4  2  1     show component encoding
#   2.16         101
```

Figure 1: SHA-256 OID in BER

```
D8 6F                               # tag(111)
   49                               # 0b010_01001: mt 2, 9 bytes
      60 86 48 01 65 03 04 02 01    # X.690 Clause 8.19
```

Figure 2: SHA-256 OID in CBOR

3.2. Encoding of a MIB Relative OID

Given some OID (e.g., "lowpanMib", assumed to be "1.3.6.1.2.1.226" [RFC7388]), to which the following is added:

```
ASN.1 Value Notation:
   { lowpanObjects(1) lowpanStats(1) lowpanOutTransmits(29) }

Dotted Decimal Notation:  .1.1.29

0D                                  # UNIVERSAL TAG 13
   03                               # 3 bytes, primitive
      01 01 1D                      # X.690 Clause 8.20
#      1  1 29                        show component encoding
```

Figure 3: MIB Relative Object Identifier in BER

```
D8 6E                               # tag(110)
   43                               # 0b010_00011: mt 2 (bstr), 3 bytes
      01 01 1D                      # X.690 Clause 8.20
```

Figure 4: MIB Relative Object Identifier in CBOR

This relative OID saves seven bytes compared to the full OID encoding.

4. Tag Factoring with Arrays and Maps

The tag content of OID tags can be byte strings (as discussed above) but also CBOR arrays and maps. The idea in the latter case is that the tag construct is factored out from each individual item in the container; the tag is placed on the array or map instead.

When the tag content of an OID tag is an array, this means that the respective tag is imputed to all elements of the array that are byte strings, arrays, or maps. (There is no effect on other elements, including text strings or tags.) For example, when the tag content of a 111 tag is an array, every array element that is a byte string is an OID, and every element that is an array or map is, in turn, treated as discussed here.

When the tag content of an OID tag is a map, this means that a tag with the same tag number is imputed to all keys in the map that are byte strings, arrays, or maps; again, there is no effect on keys of other major types. Note that there is also no effect on the values in the map.

As a result of these rules, tag factoring in nested arrays and maps is supported. For example, a 3-dimensional array of OIDs can be composed by using a single 111 tag containing an array of arrays of

arrays of byte strings. All such byte strings are then considered OIDs.

4.1. Preferred Serialization Considerations

Where tag factoring with tag number 111 is used, some OIDs enclosed in the tag may be encoded in a shorter way by using tag number 112 instead of encoding an unadorned byte string. This remains the preferred serialization (see also Section 2.2). However, this specification does not make the presence or absence of tag factoring a preferred serialization; application protocols can define where tag factoring is to be used or not (and will need to do so if they have deterministic encoding requirements).

4.2. Tag Factoring Example: X.500 Distinguished Name

Consider the X.500 distinguished name:

Attribute Types	Attribute Values
c (2.5.4.6)	US
l (2.5.4.7) s (2.5.4.8) postalCode (2.5.4.17)	Los Angeles CA 90013
street (2.5.4.9)	532 S Olive St
businessCategory (2.5.4.15) buildingName (0.9.2342.19200300.100.1.48)	Public Park Pershing Square

Table 2: Example X.500 Distinguished Name

Table 2 has four "relative distinguished names" (RDNs). The country (first) and street (third) RDNs are single valued. The second and fourth RDNs are multivalued.

The equivalent representations in CBOR diagnostic notation (Section 8 of [RFC8949]) and CBOR are:

```
111([{ h'550406': "US" },
     { h'550407': "Los Angeles",
       h'550408': "CA",
       h'550411': "90013" },
     { h'550409': "532 S Olive St" },
     { h'55040f': "Public Park",
       h'0992268993f22c640130': "Pershing Square" }])
```

Figure 5: Distinguished Name in CBOR Diagnostic Notation

```
d8 6f                                  # tag(111)
   84                                  # array(4)
      a1                               # map(1)
         43 550406                     # 2.5.4.6 (4)
         62                            # text(2)
            5553                       # "US"
      a3                               # map(3)
         43 550407                     # 2.5.4.7 (4)
         6b                            # text(11)
            4c6f7320416e67656c6573     # "Los Angeles"
         43 550408                     # 2.5.4.8 (4)
         62                            # text(2)
            4341                       # "CA"
         43 550411                     # 2.5.4.17 (4)
         65                            # text(5)
            3930303133                 # "90013"
```

```
      a1                                  # map(1)
         43 550409                        # 2.5.4.9 (4)
         6e                               # text(14)
            3533322053204f6c697665205374   # "532 S Olive St"
      a2                                  # map(2)
         43 55040f                        # 2.5.4.15 (4)
         6b                               # text(11)
            5075626c6963205061726b         # "Public Park"
         4a 0992268993f22c640130    # 0.9.2342.19200300.100.1.48 (11)
         6f                               # text(15)
            5065727368696e6720537175617265 # "Pershing Square"
```

Figure 6: Distinguished Name in CBOR (109 Bytes)

(This example encoding assumes that all attribute values are UTF-8 strings or can be represented as UTF-8 strings with no loss of information.)

5. CDDL Control Operators

Concise Data Definition Language (CDDL) specifications [RFC8610] may want to specify the use of SDNVs or SDNV sequences (as defined for the tag content for tag 110). This document introduces two new control operators that can be applied to a target value that is a byte string:

* ".sdnv", with a control type that contains unsigned integers. The byte string is specified to be encoded as an SDNV (BER encoding) [RFC6256] for the matching values of the control type.

* ".sdnvseq", with a control type that contains arrays of unsigned integers. The byte string is specified to be encoded as a sequence of SDNVs (BER encoding) [RFC6256] that decodes to an array of unsigned integers matching the control type.

* ".oid", like ".sdnvseq", except that the X*40+Y translation for absolute OIDs is included (see Figure 8).

Figure 7 shows an example for the use of ".sdnvseq" for a part of a structure using OIDs that could be used in Figure 6; Figure 8 shows the same with the ".oid" operator.

```
country-rdn = {country-oid => country-value}
country-oid = bytes .sdnvseq [85, 4, 6]
country-value = text .size 2
```

Figure 7: Using .sdnvseq

```
country-rdn = {country-oid => country-value}
country-oid = bytes .oid [2, 5, 4, 6]
country-value = text .size 2
```

Figure 8: Using .oid

Note that the control type need not be a literal; for example, "bytes .oid [2, 5, 4, *uint]" matches all OIDs inside OID arc "2.5.4", "attributeType".

6. CDDL Type Names

For the use with CDDL, the type names defined in Figure 9 are recommended:

```
oid = #6.111(bstr)
roid = #6.110(bstr)
pen = #6.112(bstr)
```

Figure 9: Recommended Type Names for CDDL

7. IANA Considerations

7.1. CBOR Tags

IANA has assigned the CBOR tag numbers in Table 3 in the 1+1 byte space (24..255) of the "CBOR Tags" registry [IANA.cbor-tags], with this document as the specification reference.

Tag	Data Item	Semantics	Reference
111	byte string, array, or map	object identifier (BER encoding)	RFC 9090
110	byte string, array, or map	relative object identifier (BER encoding); SDNV [RFC6256] sequence	RFC 9090
112	byte string, array, or map	object identifier (BER encoding), relative to 1.3.6.1.4.1	RFC 9090

Table 3: New Tag Numbers

7.2. CDDL Control Operators

IANA has assigned the CDDL control operators in Table 4 in the "CDDL Control Operators" registry [IANA.cddl], with this document as the specification reference.

Name	Reference
.sdnv	RFC 9090
.sdnvseq	RFC 9090
.oid	RFC 9090

Table 4: New CDDL Control Operators

8. Security Considerations

The security considerations of [RFC8949] apply.

The encodings in Clauses 8.19 and 8.20 of [X.690] are quite compact and unambiguous but MUST be followed precisely to avoid security pitfalls. In particular, the requirements set out in Section 2.1 of this document need to be followed; otherwise, an attacker may be able to subvert a checking process by submitting alternative representations that are later taken as the original (or even something else entirely) by another decoder that is intended to be protected by the checking process.

OIDs and relative OIDs can always be treated as opaque byte strings. Actually understanding the structure that was used for generating them is not necessary, and, except for checking the structure requirements, it is strongly NOT RECOMMENDED to perform any processing of this kind (e.g., converting into dotted notation and back) unless absolutely necessary. If the OIDs are translated into other representations, the usual security considerations for non-trivial representation conversions apply; the integer values are unlimited in range.

An attacker might trick an application into using a byte string inside a tag-factored data item, where the byte string is not actually intended to fall under one of the tags defined here. This may cause the application to emit data with semantics different from what was intended. Applications therefore need to be restrictive with respect to what data items they apply tag factoring to.

9. References

9.1. Normative References

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

[IANA.cddl]
IANA, "Concise Data Definition Language (CDDL)", <https://www.iana.org/assignments/cddl>.

[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>.

[RFC6256] Eddy, W. and E. Davies, "Using Self-Delimiting Numeric Values in Protocols", RFC 6256, DOI 10.17487/RFC6256, May 2011, <https://www.rfc-editor.org/info/rfc6256>.

[RFC8174] 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>.

[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>.

[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/info/rfc8949>.

[X.660] ITU-T, "Information technology - Procedures for the operation of object identifier registration authorities: General procedures and top arcs of the international object identifier tree", ITU-T Recommendation X.660, July 2011, <https://www.itu.int/rec/T-REC-X.660>.

[X.680] ITU-T, "Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation", ITU-T Recommendation X.680, August 2015, <https://www.itu.int/rec/T-REC-X.680>.

[X.690] ITU-T, "Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, August 2015, <https://www.itu.int/rec/T-REC-X.690>.

9.2. Informative References

[IANA.enterprise-numbers]
IANA, "Private Enterprise Numbers", <https://www.iana.org/assignments/enterprise-numbers>.

[OID-INFO] Orange SA, "Object Identifier (OID) Repository", <http://www.oid-info.com/>.

[PCRE] "PCRE - Perl Compatible Regular Expressions", <http://www.pcre.org/>.

[RFC7388] Schoenwaelder, J., Sehgal, A., Tsou, T., and C. Zhou, "Definition of Managed Objects for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 7388, DOI 10.17487/RFC7388, October 2014, <https://www.rfc-editor.org/info/rfc7388>.

[X.672] ITU-T, "Information technology - Open systems interconnection - Object identifier resolution system (ORS)", ITU-T Recommendation X.672, August 2010, <https://www.itu.int/rec/T-REC-X.672>.

Acknowledgments

Sean Leonard started the work on this document in 2014 with an elaborate proposal. Jim Schaad provided a significant review of this document. Rob Wilton's IESG review prompted us to provide preferred serialization considerations.

Contributors

Sean Leonard
Penango, Inc.
5900 Wilshire Boulevard
21st Floor
Los Angeles, CA 90036
United States of America

Email: dev+ietf@seantek.com

Author's Address

Carsten Bormann
UniversitÃ¤t Bremen TZI
Postfach 330440
D-28359 Bremen
Germany

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