Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 4 January 2021

C. Bormann
Universität Bremen TZI
S. Leonard
Penango, Inc.

3 July 2020

Concise Binary Object Representation (CBOR) Tags for Object Identifiers
draft-bormann-cbor-tags-oid-07

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.

The present document defines CBOR tags for object identifiers (OIDs). It is intended as the reference document for the IANA registration of the CBOR tags so defined.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 4 January 2021.

Copyright Notice

Table of Contents

1. Introduction 2
2. Object Identifiers 3
3. Examples 5
4. Discussion 6
5. Tag Factoring with OID Arrays and Maps 6
6. Applications and Examples of OIDs 6
7. CDDL Control Operators 8
8. IANA Considerations 9
9. Security Considerations 10
10. References 10
10.1. Normative References 11
10.2. Informative References 11
Appendix A. Change Log 12
Authors' Addresses 13

1. Introduction

The Concise Binary Object Representation (CBOR, [RFC7049]) provides for the interchange of structured data without a requirement for a pre-agreed schema. RFC 7049 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.

The present 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 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. The tags can also be applied to arrays and maps for more articulated identification purposes. It is intended as 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 RFC 7049 applies; in particular the term "byte" is used in its now customary sense as a synonym for "octet".

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 primary integer values [X.680]. 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 making use of object identifiers.

BER represents the sequence of unsigned integers by concatenating self-delimiting [RFC6256] representations of each of the primary 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 integers, the single integer 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, this specification defines two tags:

Tag TBD111: tags a byte string as the [X.690] encoding of an absolute object identifier (simply "object identifier" or "OID").

Tag TBD110: tags a byte string as the [X.690] encoding of a relative object identifier (also "relative OID"). Since the encoding of each number is the same as for [RFC6256] Self-Delimiting Numeric Values (SDNVs), this tag can also be used for tagging a byte string that contains a sequence of zero or more SDNVs.

2.1. Requirements on the byte string being tagged

A byte string tagged by TBD111 or TBD110 MUST be a syntactically valid BER representation of an object identifier: 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 must be unset; the first byte of each SDNV cannot be 0x80 (which would be a leading zero in SDNV's base-128 arithmetic).

In other words:

* its first byte, and any byte that follows a byte that has the most significant bit unset, MUST NOT be 0x80 (this requirement excludes expressing the primary integer values with anything but the shortest form)

* its last byte MUST NOT have the most significant bit set (this requirement excludes an incomplete final primary integer value)

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

[X.680] restricts RELATIVE-OID values to have 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 enable the search for specific object ID values, it is RECOMMENDED that definite length encoding (see Section 2.2.2 of [RFC7049]) is used for the byte strings 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 [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 TBD111:

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

For byte strings with tag TBD110:

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

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

3. Examples

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_01001: 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. 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.

5. Tag Factoring with OID Arrays and Maps

TBD111 and TBD110 can tag CBOR arrays and maps. The idea is that the tag is factored out from each individual byte string; the tag is placed in front of the array or map instead. The tags TBD111 and TBD110 are left-distributive.

When the TBD111 or TBD110 tag is applied to an array, it means that the respective tag is imputed to all items in the array that are byte strings. For example, when the array is tagged with TBD111, every array item that is a binary string is an OID.

When the TBD111 or TBD110 tag is applied to a map, it means that the respective tag is imputed to all keys in the map that are byte strings. The values in the map are not considered specially tagged.

Array and map nesting is permitted. For example, a 3-dimensional array of OIDs can be composed by using a single TBD111 tag, followed by an array of arrays of arrays of binary strings. All such binary strings are considered OIDs.
// That was part of the original proposal. I find it hard to imagine
// how to stop the influence of the tag deep into a nested structure.
// That's why I would rather limit this to one level (no nesting).
// But see the Figure below, which needs a nesting of two. Please
// discuss.

6. Applications and Examples of OIDs

6.1. 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 1: Example X.500 Distinguished Name

Table 1 has four "relative distinguished names" (RDNs). The country and street RDNs are single-valued. The second and fourth RDNs are multi-valued.

The equivalent representations in CBOR diagnostic notation 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.)

7. CDDL Control Operators

CDDL specifications may want to specify the use of SDNVs or SDNV sequences (as defined for the tag content for TBD110). 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 [RFC6256] SDNV (BER encoding) 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 [RFC6256] SDNVs (BER encoding) that decodes to an array of unsigned integers matching the control type.

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.

```
// We could define another control operator that includes the X*40+Y
// magic, so the example can actually use "[2, 5, 4, 6]". We could
// also add an operator that parses dotted decimal integer sequences,
// so we can use "2.5.4.6". I don't see a strong reason for that.

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

Figure 7: Using .sdnvseq

8. IANA Considerations

8.1. CBOR Tags

IANA is requested to assign the CBOR tags in Table 2, with the present document as the specification reference.

Tag	Data Item	Semantics
TBD111	multiple	object identifier (BER encoding)
TBD110	multiple	relative object identifier (BER encoding); SDNV [RFC6256] sequence

Table 2: Values for New Tags

8.2. CDDL Control Operators

IANA is requested to assign the CDDL Control Operators in Table 3, with the present document as the specification reference.

Name	Reference
.sdnv	[this document, Section 7]
.sdnvseq	[this document, Section 7]

Table 3: New CDDL Operators

9. Security Considerations

The security considerations of RFC 7049 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 supposed 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 primary integer values are unlimited in range.

9.1. Conversions Between BER and Dotted Decimal Notation

[PKILCAKE] uncovers exploit vectors for the illegal values above, as well as for cases in which conversion to or from the dotted decimal notation goes awry. Neither [X.660] nor [X.680] place an upper bound on the range of unsigned integer values for an arc; the integers are arbitrarily valued. An implementation SHOULD NOT attempt to convert each component using a fixed-size accumulator, as an attacker will certainly be able to cause the accumulator to overflow. Compact and efficient techniques for such conversions, such as the double dabble algorithm [DOUBLEDABBLE] are well-known in the art; their application to this field is left as an exercise to the reader.

10. References

10.1. Normative References

[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>.

[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>.

[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>.

[X.660] International Telecommunications Union, "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.

[X.680] International Telecommunications Union, "Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation", ITU-T Recommendation X.680, August 2015.

[X.690] International Telecommunications Union, "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.

10.2. Informative References

[DOUBLEDABBLE] Gao, S., Al-Khalili, D., and N. Chabini, "An improved BCD adder using 6-LUT FPGAs", DOI 10.1109/newcas.2012.6328944, 10th IEEE International NEWCAS Conference, June 2012, <https://doi.org/10.1109/newcas.2012.6328944>.

[OID-INFO] Orange SA, "OID Repository", 2016, <http://www.oid-info.com/>.

[PCRE] Ho, A., "PCRE - Perl Compatible Regular Expressions", 2018, <http://www.pcre.org/>.

[PKILCAKE] Kaminsky, D., Patterson, M., and L. Sassaman, "PKI Layer Cake: New Collision Attacks against the Global X.509 Infrastructure", DOI 10.1007/978-3-642-14577-3_22, Financial Cryptography and Data Security pp. 289-303, 2010, <https://doi.org/10.1007/978-3-642-14577-3_22>.

[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] International Telecommunications Union, "Information technology -- Open systems interconnection -- Object identifier resolution system", ITU-T Recommendation X.672, August 2010.

Appendix A. Change Log

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

A.1. Changes from -06 to -07

Reduce the draft back to its basic mandate: Describe CBOR tags for what is colloquially know as ASN.1 Object IDs.

A.2. Changes from -05 to -06

Refreshed the draft to the current date ("keep-alive").

A.3. Changes from -04 to -05

Discussed UUID usage in CBOR, and incorporated fixes proposed by Olivier Dubuisson, including fixes regarding OID nomenclature.

A.4. Changes from -03 to -04

Changes occurred based on limited feedback, mainly centered around the abstract and introduction, rather than substantive technical changes. These changes include:

* Changed the title so that it is about tags and techniques.

* Rewrote the abstract to describe the content more accurately, and to point out that no changes to the wire protocol are being proposed.

* Removed "ASN.1" from "object identifiers", as OIDs are independent of ASN.1.

* Rewrote the introduction to be more about the present text.

* Proposed a concise OID arc.

* Provided binary regular expression forms for OID validation.

* Updated IANA registration tables.

A.5. Changes from -02 to -03

Many significant changes occurred in this version. These changes include:

* Expanded the draft scope to be a comprehensive CBOR update.

* Added OID-related sections: OID Enumerations, OID Maps and Arrays, and Applications and Examples of OIDs.

* Added Tag 36 update (binary MIME, better definitions).

* Added stub/experimental sections for X.690 Series Tags (tag <<X>>) and Regular Expressions (tag 35).

* Added technique for representing sets and multisets.

* Added references and fixed typos.

Authors' Addresses

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

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

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

Email: dev+ietf@seantek.com
URI: http://www.penango.com/