CBOR Working Group M. Richardson
Internet-Draft Sandelman Software Works
Intended status: Standards Track C. Bormann
Expires: 25 April 2022 Universität Bremen TZI
22 October 2021

CBOR tags for IPv4 and IPv6 addresses and prefixes
draft-ietf-cbor-network-addresses-13

Abstract

This specification defines two CBOR Tags for use with IPv6 and IPv4 addresses and prefixes.

// RFC-EDITOR-please-remove: This work is tracked at
// https://github.com/cbor-wg/cbor-network-address

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

[RFC8949] defines a number of CBOR Tags for common items. Tags 260 and 261 were later defined in drafts listed with IANA [IANA.cbor-tags]. These tags were intended to cover addresses (260) and prefixes (261). Tag 260 distinguishes between IPv6, IPv4, and MAC [RFC7042] addresses only through the length of the byte string, making it impossible, for example, to drop trailing zeros in the encoding of IP addresses. Tag 261 was not documented well enough for use.

This specification defines tags 54 and 52 achieving an explicit indication of IPv6 or IPv4 by the tag number. These new tags are intended to be used in preference to tags 260 and 261. They provide formats for IPv6 and IPv4 addresses, prefixes, and addresses with prefixes, achieving an explicit indication of IPv6 or IPv4. The prefix format omits trailing zeroes in the address part. (Due to the complexity of testing, the value of omitting trailing zeros for the pure address format was considered non-essential and support for that is not provided in this specification.) This specification does not deal with MAC addresses (Section 2 of [RFC7042]) such as they are used for Ethernet.

2. 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.

3. Protocol

3.1. Three Forms

3.1.1. Addresses

These tags can be applied to byte strings to represent a single address.

This form is called the Address Format.

3.1.2. Prefixes

When applied to an array that starts with an unsigned integer, they represent a CIDR-style prefix of that length.

When the Address Format (i.e., without prefix) appears in a context where a prefix is expected, then it is to be assumed that all bits are relevant. That is, for IPv4, a /32 is implied, and for IPv6, a /128 is implied.

This form is called the Prefix Format.

3.1.3. Interface Definition

When applied to an array that starts with a byte string, which stands for an IP address, followed by an unsigned integer giving the bit length of a prefix built out of the first length bits of the address, they represent information that is commonly used to specify both the network prefix and the IP address of an interface.

The length of the byte string is always 16 bytes (for IPv6) and 4 bytes (for IPv4).

This form is called the Interface Format.

Interface Format definitions support an optional third element to the array, which is to be used as the IPv6 Link-Local zone identifier from Section 6 of [RFC4007]; for symmetry this is also provided for IPv4 as in [RFC4001] and [RFC6991]. The zone identifier may be an integer, in which case it is to be interpreted as the interface index. It may be a text string, in which case it is to be interpreted as an interface name.

As explained in [RFC4007] the zone identifiers are strictly local to the node. They are useful for communications within a node about connected addresses (for instance, where a link-local peer is discovered by one daemon, and another daemon needs to be informed). They may also have utility in some management protocols.

In the cases where the Interface Format is being used to represent only an address with a zone identifier, and no interface prefix information, then the prefix length may be replaced with the CBOR "null" (0xF6).

3.2. IPv6

IANA has allocated tag 54 for IPv6 uses. (This is the ASCII code for '6'.)

An IPv6 address is to be encoded as a sixteen-byte byte string (Section 3.1 of [RFC8949], major type 2), enclosed in Tag number 54.

For example:

```
54(h'20010db81234deedbeefcafefacefeed')
```

An IPv6 prefix, such as 2001:db8:1234::/48 is to be encoded as a two element array, with the length of the prefix first. See Section 4 for the detailed construction of the second element.

For example:

```
54([48, h'20010db81234'])
```

An IPv6 address combined with a prefix length, such as being used for configuring an interface, is to be encoded as a two element array, with the (full-length) IPv6 address first and the length of the associated network the prefix next; a third element can be added for the zone identifier.

For example:

```
54([h'20010db81234deedbeefcafefacefeed', 56])
```

The address-with-prefix form can be reliably distinguished from the prefix form only in the sequence of the array elements.

Some example of a link-local IPv6 address with a 64-bit prefix:

```
54([h'fe8000000000020202fffffffe030303', 64, 'eth0'])
```

with a numeric zone identifier:

```
54([h'fe8000000000020202fffffffe030303', 64, 42])
```

An IPv6 link-local address without a prefix length:

```
54([h'fe8000000000020202fffffffe030303', null, 42])
```

Zone identifiers may be used with any kind of IP address, not just Link-Local addresses. In particular, they are valid for multicast addresses, and there may still be some significance for Globally Unique Addresses (GUA).

3.3. IPv4

IANA has allocated tag 52 for IPv4 uses. (This is the ASCII code for '4'.)

An IPv4 address is to be encoded as a four-byte byte string (Section 3.1 of [RFC8949], major type 2), enclosed in Tag number 52.

For example:

```
52(h'c0000201')
```

An IPv4 prefix, such as 192.0.2.0/24 is to be encoded as a two element array, with the length of the prefix first. See Section 4 for the detailed construction of the second element.

For example:

```
52([24, h'c00002'])
```

An IPv4 address combined with a prefix length, such as being used for configuring an interface, is to be encoded as a two element array, with the (full-length) IPv4 address first and the length of the associated network the prefix next; a third element can be added for the zone identifier.

For example, 192.0.2.1/24 is to be encoded as a two element array, with the length of the prefix (implied 192.0.2.0/24) last.

```
52([h'c0000201', 24])
```

The address-with-prefix form can be reliably distinguished from the prefix form only in the sequence of the array elements.

4. Tag validity

This section discusses when a tag 54 or tag 52 is valid (Section 5.3.2 of [RFC8949]). As with all CBOR tags, validity checking can be handled in a generic CBOR library or in the application. A generic CBOR library needs to document whether and how it handles validity checking.

The rule ip-address-or-prefix in Figure 1 shows how to check the overall structure of these tags and their content, the ranges of integer values, and the lengths of byte strings. An instance of tag 52 or 54 is valid if it matches that rule and, for ipv6-prefix and ipv4-prefix, the considerations of Sections 4.2 and 4.3.

4.1. Deterministic Encoding

The tag validity rules, combined with the rules in Section 4.2.1 of [RFC8949], lead to deterministic encoding for tags 54 and 52 and require no further Additional Deterministic Encoding Considerations as per Section 4.2.2 of [RFC8949].

4.2. Encoder Considerations for Prefixes

For the byte strings used as the second element in the array representing a prefix:

(1) An encoder MUST set any unused bytes, and any unused bits in the final byte, if any, to zero. Unused bytes/bits are bytes/bits that are not covered by the prefix length given. So for example, 2001:db8:1230::/44 MUST be encoded as:

```
54([44, h'20010db81230'])
```

even though variations like:

```
54([44, h'20010db81233'])
54([44, h'20010db8123f'])
54([44, h'20010db8123012'])
```

start with the same 44 bits, but are not valid.

(Analogous examples can be constructed for IPv4 prefixes.)

(2) An encoder MUST then omit any right-aligned (trailing) sequence of bytes that are all zero.

There is no relationship between the number of bytes omitted and the prefix length. For instance, the prefix 2001:db8::/64 is encoded as:

```
54([64, h'20010db8'])
```

4.3. Decoder Considerations for Prefixes

A decoder MUST check that all unused bits encoded in the byte string ipv6-prefix-bytes/ipv4-prefix-bytes, i.e., the bits to the right of the prefix length, are zero.

A decoder MUST also check that the byte string does not end in a zero byte.

Since encoders are required to remove zero-valued trailing bytes, a decoder MUST handle the case where a prefix length specifies that more bits are relevant than are actually present in the byte-string.

As an example, ::/128 is encoded as

```
54([128, h''])
```

4.3.1. Example implementation

A recommendation for prefix decoder implementations is to first create an array of 16 (or 4) zero bytes.

Then taking whichever is smaller between (a) the length of the included byte-string, and (b) the number of bytes covered by the prefix-length rounded up to the next multiple of 8: fail if that number is greater than 16 (or 4), and then copy that many bytes from the byte-string into the byte array.

Finally, looking at the number of unused bits in the last byte (if any) of the range covered by the prefix length, check that any unused bits in the byte string are zero:

```
unused_bits = (8 - (prefix_length_in_bits % 8)) % 8;
if (length_in_bytes > 0 &&
    (address_bytes[length_in_bytes - 1] & ~(0xFF << unused_bits))
       != 0)
  fail();
```

5. CDDL

For use with CDDL [RFC8610], the typenames defined in Figure 1 are recommended:

```
ip-address-or-prefix = ipv6-address-or-prefix /
                       ipv4-address-or-prefix

ipv6-address-or-prefix = #6.54(ipv6-address /
                               ipv6-address-with-prefix /
                               ipv6-prefix)
ipv4-address-or-prefix = #6.52(ipv4-address /
                               ipv4-address-with-prefix /
                               ipv4-prefix)

ipv6-address = bytes .size 16
ipv4-address = bytes .size 4

ipv6-address-with-prefix = [ipv6-address,
                            ipv6-prefix-length / null,
                            ?ip-zone-identifier]
ipv4-address-with-prefix = [ipv4-address,
                            ipv4-prefix-length / null,
                            ?ip-zone-identifier]

ipv6-prefix-length = 0..128
ipv4-prefix-length = 0..32

ipv6-prefix = [ipv6-prefix-length, ipv6-prefix-bytes]
ipv4-prefix = [ipv4-prefix-length, ipv4-prefix-bytes]

ipv6-prefix-bytes = bytes .size (uint .le 16)
ipv4-prefix-bytes = bytes .size (uint .le 4)

ip-zone-identifier = uint / text
```

Figure 1: CDDL types for tags 54 and 52

6. Security Considerations

This document provides an CBOR encoding for IPv4 and IPv6 address information. Any applications using these encodings will need to consider the security implications of these data in their specific context. For example, identifying which byte sequences in a protocol are addresses may allow an attacker or eavesdropper to better understand what parts of a packet to attack.

Applications need to check the validity (Section 4) of a tag before acting on any of its contents. If the validity checking is not done in the generic CBOR decoder, it needs to be done in the application; in any case it needs to be done before the tag is transformed into a platform-specific representation that could conceal validity errors.

The right-hand bits of the prefix, after the prefix-length, are set to zero by this protocol. (Otherwise, a malicious party could use them to transmit covert data in a way that would not affect the primary use of this encoding. Such abuse is detected by tag validity checking, and can also be detected by examination of the raw protocol bytes.)

7. IANA Considerations

IANA has allocated two tags from the Specification Required area of the Concise Binary Object Representation (CBOR) Tags [IANA.cbor-tags]:

7.1. Tag 54 - IPv6

Data Item: byte string or array
Semantics: IPv6, [prefixlen,IPv6], [IPv6,prefixpart]

7.2. Tag 52 - IPv4

Data Item: byte string or array
Semantics: IPv4, [prefixlen,IPv4], [IPv4,prefixpart]

7.3. Tags 260 and 261

IANA is requested to add the note "DEPRECATED in favor of 52 and 54 for IP addresses" to registrations 260 and 261

8. References

8.1. 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>.

[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>.

8.2. Informative References

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

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, DOI 10.17487/RFC4001, February 2005, <https://www.rfc-editor.org/info/rfc4001>.

[RFC4007] Deering, S., Haberman, B., Jinmei, T., Nordmark, E., and B. Zill, "IPv6 Scoped Address Architecture", RFC 4007, DOI 10.17487/RFC4007, March 2005, <https://www.rfc-editor.org/info/rfc4007>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7042] Eastlake 3rd, D. and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", BCP 141, RFC 7042, DOI 10.17487/RFC7042, October 2013, <https://www.rfc-editor.org/info/rfc7042>.

Appendix A. Changelog

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

* 03

* 02

* 01 added security considerations about covert channel

Acknowledgements

Roman Danyliw, Donald Eastlake, Ben Kaduk, Barry Leiba, and Éric Vyncke reviewed the document and provided suggested text. Jürgen Schönwälder helped finding the history of IPv4 zone identifiers.

Authors' Addresses

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

Carsten Bormann
Universität Bremen TZI
Germany

Email: cabo@tzi.org