Workgroup: CBOR Working Group
Internet-Draft:
draft-ietf-cbor-network-addresses-01
Published: 7 March 2021
Intended Status: Standards Track
Expires: 8 September 2021
Authors: M. Richardson
Sandelman Software Works

# CBOR tags for IPv4 and IPv6 addresses and prefixes

## Abstract

This document describes two CBOR Tags to be used with IPv4 and IPv6 addresses and prefixes.

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

## 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 8 September 2021.

**Table of Contents**



## 1. Introduction

[RFC8949] defines a number of CBOR Tags for common items.

Not included are ones to indicate if the item is an IPv4 or IPv6 address, or if it is an address plus prefix length. This document defines them.

## 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

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

When applied to an array, the represent a CIDR-style prefix. When a byte string (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.

### 3.1. IPv6

IANA has allocated tag TBD1 for IPv6 uses.

An IPv6 address is to be encoded as up to sixteen-byte byte string ([RFC8949] section, 3.1, major type 2), prefixed with tag TBD1. Trailing zero octets MAY be omitted.

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:

```
TBD1([ 48, h'20010db81234'])
```

### 3.2. IPv4

IANA has allocated tag TBD2 for IPv4 uses.

An IPv4 address is to be encoded as a four-byte byte string ([RFC8949] section, 3.1, major type 2), prefixed with tag TBD2. Trailing zero octets MAY be omitted.

An IPv4 prefix, such as 192.0.2.1/24 is to be encoded as a two element array, with the length of the prefix first:

```
 TBD2([ 24, h'C0000201'])
```

## 4. Encoder Consideration for prefixes

An encoder may omit as many right-hand (trailing) bytes which are all zero as it wishes.

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

```
TBD1([64, h'20010db8'])
```

An encoder MUST take care to set all trailing bits to zero. While decoders are expected to ignore them, such garbage entities could be used as a covert channel, or may reveal the state of what would otherwise be private memory contents. So for example, 2001:db8:1230::/44 MUST be encoded as:

```
TBD1([44, h'20010db81230'])
```

even though variations like:

```
TBD1([44, h'20010db81233'])  WRONG
TBD1([45, h'20010db8123f'])  WRONG
```

would be parsed in the exact same way.

The same considerations apply to IPv4 prefixes.

## 5. Decoder Considerations for prefixes

A decoder MUST consider all bits to the right of the prefix length to be zero.

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 a pathological case, ::/128 can be encoded as

```
TBD1([0, h''])
```

(EDNOTE: do we want to support:

```
[0]
```

or

```
[0, null]
```

(EDNOTE: what if the array has more than 2 members? Is this a convert channel, or is this a possible extension point?)

A recommendation for implementation is to first create an array of 16 (or 4) bytes in size, set it all to zero.

Then looking at the length of the included byte-string, and of the prefix-length, rounded up to the next multiple of 8, and taking whichever is smaller, copy that many bytes from the byte-string into the array.

Finally, looking at the last three bits of the prefix-length (that is, the prefix-length modulo 8), use a static array of 8 values to force the lower bits, non-relevant bits to zero.

A particularly paranoid decoder could examine the lower non-relevant bits to determine if they are non-zero, and reject the prefix. This would detect non-compliant encoders, or a possible covert channel.

## 6. Security Considerations

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.

Reading the relevant RFC may provide more information, so it would seem that any additional security that was provided by not being able to identify what are IP addresses falls into the security by obscurity category.

The right-hand bits of the prefix, after the prefix-length, are ignored by this protocol. 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 would be detected by examination of the raw protocol bytes. Users of this encoding should be aware of this possibility.

## 7. IANA Considerations

IANA is asked to allocate two tags from the Specification Required area of the Concise Binary Object Representation (CBOR) Tags, in the ("1+1") area.

### 7.1. TBD1 - IPv6

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

### 7.2. TBD2 - IPv4

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

## 8. Acknowledgements

none yet

## 9. Changelog

*01 added security considerations about covert channel

## 10. Normative References

**[BCP14]** 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>.

**[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>.

**[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>.

## Author's Address

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca