Using CDDL to Model Weave TLV Structured Data

Technical Specification

Revision 0.3
2017-09-06

Abstract

When modeling structured data encoded in the Weave TLV Format [WTLV], having a well-defined data modeling language is helpful. A suitable structured data modeling language designed especially for the Weave TLV Format need not be invented, because a suitable one already exists: the CBOR Data Definition Language (CDDL) [ID.CDDL]. This document presents a congruency between all possible texts encoded in the Weave TLV Format and a contrived encoding of those texts in Concise Binary Object Representation (CBOR) [RFC7049]. Accordingly, this document also specifies a CDDL preamble providing type and group definitions for use in modeling texts so encoded, and which may therefore be used in modeling structured data for presentation in the native Weave TLV Format.

Introduction

The Weave TLV Format [WTLV] is a concise structured data interchange language with the following basic features:

- Portable representations for elements of primitive type, e.g. booleans, fixed width machine integers, IEEE 754 floating point numbers, octet sequences and text strings encoded with UTF-8.
- Three types of container element: array, map and path.
- Attribute tags applied to each primitive and container element.

This document describes a canonical translation for all documents encoded in Weave TLV Format to and from their corresponding encoding as documents in the Concise Binary Object Representation (CBOR) [RFC7049]. Additionally, definitions are further provided in the CBOR Data Definition Language [ID.CDDL] for use in modeling structured data for interchange with the Weave TLV Format.

Table Of Contents

Overview

Every text encoded in the Weave TLV Format comprises an unbounded sequence of one or more Weave TLV *elements*. Logically, an element consists of an optional *tag* and a *value* annotated by its *element type*, which is either a *primitive* type or a *container* type when it comprises a sequence of zero or more contained elements.

Encoding the Weave TLV Format in CBOR is straightforward. For each element type, the corresponding CBOR representation (major and minor numbers) is identified. Some new CBOR tags are defined, one for each of the categories of Weave TLV tag: *profile-specific* and *context-specific*. Elements using the *anonymous* Weave TLV tag are encoded as untagged CBOR items. An additional CBOR tag is defined for distinguishing *path* container elements from *array* container elements, which are constrained to contain only anonymous elements.

Finally, a new CBOR tag is defined for identifying CBOR byte string items (major type 2) that are specifically encoded in the Weave TLV Format. A formal request is additionally made to IANA for this tag to be registered in the Concise Binary Object Representation (CBOR) Tags registry [CBORTAGS].

Detailed Requirements

In this section, the translation of Weave TLV Format elements into their corresponding CBOR items is presented in three sections. First, the CBOR representations of the various Weave TLV element types are defined (using the CBOR Data Definition Language [ID.CDDL]). Next, the new CBOR tags corresponding to the Weave TLV tag categories are defined.

Primitive Element Types

The following table lists the Weave TLV Format primitive element types and their corresponding CBOR representation written as a CDDL type.

Type Code	*Type Description*	*CDDL Type*
`00000`	Signed integer, 1 octet	`int .size 1`
`00001`	Signed integer, 2 octets	`int .size 2`
`00010`	Signed integer, 4 octets	`int .size 4`
`00011`	Signed integer, 8 octets	`int .size 8`

00100	Unsigned integer, 1 octet	uint .size 1
00101	Unsigned integer, 2 octets	uint .size 2
00110	Unsigned integer, 4 octets	uint .size 4
00111	Unsigned integer, 8 octets	uint .size 8
01000	Boolean false	false
01001	Boolean true	true
01010	Floating point number, 4 octets	float32
01011	Floating point number, 8 octets	float64
01100	UTF-8 text, 1 octet length	tstr .size (0..0xff)
01101	UTF-8 text, 2 octet length	tstr .size (0..0xffff)
01110	UTF-8 text, 4 octet length	tstr .size (0..0xffffffff)
01111	UTF-8 text, 8 octet length	tstr .size (0..0xffffffffffffffff)
10000	Octet sequence, 1 octet length	bstr .size (0..0xff)
10001	Octet sequence, 2 octet length	bstr .size (0..0xffff)
10010	Octet sequence, 4 octet length	bstr .size (0..0xffffffff)
10011	Octet sequence, 8 octet length	bstr .size (0..0xffffffffffffffff)
10100	Null	nil

The indefinite length octet sequence and text string minor types are not used in the CBOR representation of Weave TLV.

Tag Categories

The tag control field in the encoding of elements in the Weave TLV Format indicates both the category of the tag and the type of the value encoded in the tag. Four new CBOR tag values are defined for used in encoding Weave TLV tag control field values, as follows:

- Tag <*X*> (value TBD): associates a Context-specific tag number with a Weave TLV value.

- Tag <*C*> (value TBD): associates a Profile-specific tag number for the Weave Common Profile with a Weave TLV value.

- Tag <I> (value TBD): associates a Profile-specific tag number for the profile with the *vendor id* and *profile id* that is implicitly associated with the context of the encoded text with a Weave TLV value.

- Tag <Q> (value TBD): associates a Profile-specific tag number for the profile explicitly listed in the Weave Profile Registry by its *vendor id* and *profile id* with a Weave TLV value.

Tag Control	*Tag Description*	*CDDL Type*
`000`	Anonymous	Not defined.
`001`	Context-specific	`#6.<X>(uint .size 1)`
`010`	Common profile, 2 octets	`#6.<C>(uint .size 2)`
`011`	Common profile, 4 octets	`#6.<C>(uint .size 4)`
`100`	Implicit profile, 2 octets	`#6.<I>(uint .size 2)`
`101`	Implicit profile, 4 octets	`#6.<I>(uint .size 4)`

`110`	Fully-qualified profile, 6 octets	`#6.<Q>([`
		`  uint .size 2,`
		`  uint .size 2,`
		`  uint .size 2`
		`])`

`111`	Fully-qualified profile, 8 octets	`#6.<Q>([`
		`  uint .size 2,`
		`  uint .size 2,`
		`  uint .size 4`
		`])`

Anonymous elements in Weave TLV are encoded in CBOR as untagged items. The CDDL types in each of the remaining rows of the table above are collected into a type named `wtlv-tag` that comprises the choice.

Container Types

The following table lists the Weave TLV Format container element types and their corresponding CBOR representation written as a CDDL type, where the group `wtlv-element` represents any Weave TLV element (tagged or untagged), the type `wtlv-tag` is any Weave TLV tag value, and `wtlv-anon` is any untagged Weave TLV element.

The CDDL definition of the `wtlv-element` group follows:

```
wtlv-element = $$wtlv-element

$$wtlv-element //= wtlv-anon
$$wtlv-element //= ( wtlv-tag => wtlv-anon )
```

Additionally, a CBOR tag is defined for use in identifying arrays (major type 4) that represent Weave TLV path elements.

- Tag <*S*> (value TBD): identifies an array of elements encapsulated in a Weave TLV Format path element.

This definition is used in the following table of container type definitions:

Type Code	*Type Description*	*CDDL Type*
`10101`	Structure	`{ * wtlv-tag => wtlv-anon }`
`10110`	Array	`[ * wtlv-anon ]`
`10111`	Path	`#6.<S>([ * $$wtlv-element ])`
`11000`	End of container	`#7.31`

Finally the `wtlv-anon` type is defined as the choice of all untagged primitive and container types from the tables above.

Note well: the *End of container* element type in the Weave TLV Format is functionally equivalent to the *Break* simple value in CBOR. It is used to mark the end of an array or map of indefinite length in the same way that container types are terminated in the Weave TLV Format by the presence of an *End of Container* element.

Reserved Element Types

The element type codes `11001` to `11111` were not defined in Weave TLV when this specification was drafted. Their translation into a CBOR representation is accordingly not yet defined.

Embedding Weave TLV In CBOR Byte Strings

To facilitate the inclusion of Weave TLV Format data in CBOR byte string (major type 4) items, this section defines an appropriate CBOR tag.

- Tag <*W*> (value TBD): identifies a byte string encoded in Weave TLV Format.

IANA is requested to assign tag <*W*> to any available code point in the First Come First Served range 256 to 18446744073709551615 with a *Data item* indicating major type 4 and Semantics indicating "Weave TLV Format" encoding.

CDDL Preamble

The following addition to the CDDL preamble is defined for use in modeling structured data encoded in the Weave TLV Format, as well as in the corresponding translation of such data into its equivalent CBOR representation.

```
; Tag number
wtlv-tag-num = uint .size 4

; Tags with only a tag number
wtlv-tag-common = #6.<C>(wtlv-tag-num)   ; Weave Common Profile tag
wtlv-tag-implicit = #6.<I>(wtlv-tag-num) ; Implicit profile-specific tag
wtlv-tag-context = #6.<X>(wtlv-tag-num)  ; Context-specific tag

; Tag qualified by vendor identifier and profile number
wtlv-tag-qualified = #6.<Q>([
      uint .size 2,     ; Weave Profile Vendor Identifier
      uint .size 2,     ; Weave Profile Number
      wtlv-tag-num,     ; Registered tag number
])

; Tags
wtlv-tag = (
      wtlv-tag-common / wtlv-tag-implicit / wtlv-tag-context /
      wtlv-tag-qualified
)

; Forward declarations of sockets for element group and anonymous type
wtlv-element = $$wtlv-element
wtlv-anon = $wtlv-anon

; Elements (anonymous and tagged values)
$$wtlv-element //= ( wtlv-anon )
$$wtlv-element //= ( wtlv-tag => wtlv-anon )

; Values (primitives and containers)
$wtlv-anon /= nil
$wtlv-anon /= bool
$wtlv-anon /= int .size 8
$wtlv-anon /= uint .size 8
```

```
$wtlv-anon /= float32_64
$wtlv-anon /= bstr .size (0..0xffffffffffffffff)
$wtlv-anon /= tstr .size (0..0xffffffffffffffff)
$wtlv-anon /= { * wtlv-tag => wtlv-anon }
$wtlv-anon /= [ * wtlv-anon ]
$wtlv-anon /= #6.<S>([ * wtlv-element ])

; Tag generics (constructor functions)
wtlv-tag-mkc<n> = #6.<C>(n) .within wtlv-tag-common
wtlv-tag-mki<n> = #6.<I>(n) .within wtlv-tag-implicit
wtlv-tag-mkq<n> = #6.<Q>(n) .within wtlv-tag-qualified
wtlv-tag-mkx<n> = #6.<X>(n) .within wtlv-tag-context
wtlv-tag-mks<s> = #6.<S>(s) .within #6.<S>([ * $$wtlv-element ])
```

Editor's Note: The tokens <*C*>, <*I*>, <*Q*>, <*S*> and <*X*> in the preceding table need not actually be defined by IANA in its Concise Binary Object Representation (CBOR) Tags [CBORTAGS] registry. They are only used in this document for illustrative purposes.

Note the use of the generic types used for constructing tags of the various categories, i.e. the `wtlv-tag-mkc<n>`, `wtlv-tag-mki<n>`, `wtlv-tag-mkq<n>` and `wtlv-tag-mkx<n>` definitions. These are used in the examples below to define constant tag values in the CDDL provided to demonstrate how to model data for interchange in the Weave TLV Format.

As an initial example, the following CDDL excerpt is a model of a Weave TLV data structure commonly used in the Weave Schema [SCHEMA], i.e. an element that is either a A) UTF-8 text string of maximum length 256 octets, or B) an unsigned 64-bit integer treated as a reference to a well-known string value.

```
WeaveStringRef = uint .size 8
WeaveString256 = tstr .size (0..0xff)

WeaveString = WeaveStringRef / WeaveString256
```

Using the `WeaveString` data model shown in the example above, the following CDDL excerpt is a model of the Weave TLV data structure comprising the Weave Device Identity Trait [DEV.ID].

```
Vendor-nestlabs = 0x235A      ; Nest Labs, Inc. vendor identifier
Profile-device-trait = 0x0017 ; Device Identity Trait profile number

; Fully-qualified tag of path to properties
Tag-trait = wtlv-tag-mkq<[Vendor-nestlabs, Profile-device-trait, 0 ]>

; Context-specific tags
Tag-vendor-id           = wtlv-tag-mkx<1>
Tag-vendor-id-desc= wtlv-tag-mkx<2>
Tag-product-id          = wtlv-tag-mkx<3>
```

```
Tag-product-id-desc     = wtlv-tag-mkx<4>
Tag-product-rev         = wtlv-tag-mkx<5>
Tag-serial-num          = wtlv-tag-mkx<6>
Tag-sw-version          = wtlv-tag-mkx<7>
Tag-manufacture-date    = wtlv-tag-mkx<8>

; Property types
Prop-vendor-id          = ( Tag-vendor-id => 1..65534 )
Prop-vendor-id-desc     = ( Tag-vendor-id-desc => WeaveString )
Prop-product-id         = ( Tag-product-id => 1..65534 )
Prop-product-id-desc    = ( Tag-product-id-desc => WeaveString )
Prop-product-rev        = ( Tag-product-rev => uint .size 2 )
Prop-serial-num         = ( Tag-serial-num => tstr .size 32 )
Prop-sw-version         = ( Tag-sw-version => tstr .size 32 )
Prop-manufacture-date   = ( Tag-manufacture-date => tstr .size 32 )

Properties = {
      Prop-vendor-id,
      ? Prop-vendor-id-desc,
      Prop-product-id,
      ? Prop-product-id-desc,
      Prop-product-rev,
      Prop-serial-num,
      Prop-sw-version,
      ? Prop-manufacture-date,      ; constraint: ISO-8601 formatted
}

Trait = [ Tag-trait => wtlv-tag-mks<[ Properties ]> ]
```

In the example above, the Device Identity trait is a fully-qualified profile-specific tag identifying the Device Identity Profile Number (0x0017) in the Nest Labs Vendor Identity (0x235A) associated to a Weave TLV Format path element containing one encapsulated element, which is the structure of the properties defined in the trait, some of which are optional and some of which are required, each with the type defined for it accordingly.

Editor's Note: it would probably be better to define the type of the Manufacturing Date property to have the `tdate .size 32` type, except the `wtlv-anon` type defined in the preamble above does not actually admit UTF-8 text strings tagged with non-Weave CBOR tags. Not sure how to deal with that issue. The Weave TLV Format doesn't actually have a way to identify that a text string has a particular format like CBOR does, and the encoding defined in this document therefore doesn't try to add one.

Comparisons

This section provides a comparison of the octet sequences produced by encoding data in the Weave TLV Format and by encoding the same date in its corresponding translation into CBOR.

Editor's note: to facilitate this illustration, it is necessary to assign temporary code points for the <*C*>, <*I*>, <*S*>, <*Q*> and <*X*> CBOR tags introduced in this document without any official registration by IANA. The <*C*>, <*I*>, <*Q*> and <*X*> tags are commonly used in the Weave TLV translation to CBOR, so the optimal case would be for these to be assigned from the Standards Action Required range 0 to 23, so they could be encoded in the minor number of major type 0. That's more than a little optimistic, but assuming it's possible simplifies the illustration.

The new CBOR tags defined in this document are assigned temporary values in the following table solely to facilitate the illustration in this section:

Tag	*CDDL Representation Type*	*Semantic Description*
6	`wtlv-tag-num`	<*C*> Common profile-specific tag number.
7	`wtlv-tag-num`	<*I*> Implicit profile-specific tag number.

9	`[`
	`  uint .size 2,`
	`  uint .size 2,`
	`  wtlv-tag-num`
	`]`
		<*Q*> Fully-qualified profile-specific tag number.

8	`wtlv-tag-num`	<*X*> Context-specific tag number.
95	`[* wtlv-element ]`	<*S*> Path container.

The following sections present examples of data structures encoded first in the Weave TLV Format and second in the CBOR translation of Weave TLV structured data presented in this document.

Device Identity Trait

In this section, the octet sequence encoding the Device Identity Trait for a Nest Learning Thermostat (Third Generation) in Weave TLV Format is compared with the octet sequence encoding the same information in the CBOR translation of Weave TLV defined above.

Nest Learning Thermostat (Third Generation) in Weave TLV Format

The octet sequence representing the Device Identity Trait for the Nest Learning Thermostat (Third Generation) is typically about 39 octets in length. The following table presents an annotated description of each Weave TLV Format element.

Element Name	*Octets*	*Weave TLV Annotated Encoding*
Vendor identifier	`25 01 5a 23`	Control: context-tag, unsigned 16-bit Tag: 1 Value: 0x235A
Product identifier	`24 02 0a`	Control: context-tag, unsigned 8-bit Tag: 2 Value: 0xA
Product revision	`24 03 01`	Control: context-tag, unsigned 8-bit Tag: 3 Value: 1
Serial Number	`2c 06 10 30 39 41 41 30 31 41 43 43 33 31 35 30 5a 44 45`	Control: context-tag, UTF-8 string, length 16 Tag: 6 Value: “09AA01AC33150ZDE”
Software Version	`2c 07 07 35 2e 31 2e 38 2d 33`	Control: context-tag, UTF-8 string, length 7 Tag: 7 Value: “5.1.8-3”

For comparison, the same data would be encoded in 40 octets with the CBOR translation of the Weave TLV Format as shown in the following similar table:

Element Name	*Octets*	*CBOR Diagnostic Encoding*
Vendor identifier	`C8 01 19 23 5a`	`8(1) 0x235A`
Product identifier	`C8 02 0a`	`8(2) 0x000A`
Product revision	`C8 03 01`	`8(3) 0x0001`
Serial Number	`C8 06 70 30 39 41 41 30 31 41 43 43 33 31 35 30 5a 44 45`	`8(6)` “09AA01AC33150ZDE”

Software Version	C8 07 67 35 2e 31 2e 38 2d 33	8(7) “5.1.8-3”

Editor’s note: the length of the translated encoding would be longer if the code points assigned to the <C>, <I>, <Q> and <X> CBOR tags were chosen from the Specification Required range instead of the Standards Action range. Instead of encoding tag numbers for each element with two octets, it would require three octets. If chosen from the First Come First Served range, it would require at least four.

References

Normative References

ID.CDDL	C. Vigano and H. Berkholz, “CBOR data definition language (CDDL): a notational convention to express CBOR data structures”, I-D.greevenbosch-appsawg-cbor-cddl-10, March 2017.
RFC7049	C. Bormann and P. Hoffman, “Concise Binary Object Representation (CBOR)”, RFC 7049, October 2013
WTLV	Weave TLV, Specification

Informative References

CBORTAGS	Concise Binary Object Representation (CBOR) Tags, IANA registry.
DEV.ID	Weave: Device Identity Trait: Design Specification
SCHEMA	Weave Schema Guide, v0.3