Internet Engineering Task Force (IETF)
Request for Comments: 9254
Category: Standards Track
ISSN: 2070-1721

M. Veillette, Ed.
Trilliant Networks Inc.
I. Petrov, Ed.
Google Switzerland GmbH
A. Pelov
Acklio
C. Bormann
Universität Bremen TZI
M. Richardson
Sandelman Software Works
July 2022

Encoding of Data Modeled with YANG in the Concise Binary Object Representation (CBOR)

Кодирование данных модели YANG в CBOR

Аннотация

Язык моделирования данных YANG (RFC 7950) служит для моделирования данных конфигурации и состояния, параметров и результатов вызовов удалённых процедур (Remote Procedure Call или RPC) или действий и уведомлений.

Этот документ определяет представление правил YANG в CBOR (Concise Binary Object Representation) (RFC 8949).

Статус документа

Документ относится к категории Internet Standards Track.

Документ является результатом работы IETF[1] и представляет согласованный взгляд сообщества IETF. Документ прошёл открытое обсуждение и был одобрен для публикации IESG[2]. Дополнительную информацию о стандартах Internet можно найти в разделе 2 в RFC 7841.

Информация о текущем статусе документа, найденных ошибках и способах обратной связи доступна по ссылке https://www.rfc-editor.org/info/rfc9254.

Оглавление

[1]Internet Engineering Task Force - комиссия по решению инженерных задач Internet.
[2]Internet Engineering Steering Group - комиссия по инженерным разработкам Internet.

1. Введение

Спецификация языка моделирования данных YANG 1.1 [RFC7950] определяет представление XML для экземпляров данных, т. е. содержимого хранилищ конфигурации, данных состояния, ввода и вывода RPC и действий, уведомлений о событиях. Дополнительный набор правил кодирования задан в [RFC7951] на основе The JavaScript Object Notation (JSON) Data Interchange Format" [RFC8259].

Целью этого документа является задание набора правил кодирования для краткого представления двоичных объектов (Concise Binary Object Representation или CBOR) [RFC8949], называемого YANG-CBOR. Задаваемое кодирование компактней XML и JSON и более подходит для узлов и/или сетей с ограничениями, описанных в [RFC7228].

2. Терминология и нотация

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не нужно** (SHALL NOT), **следует** (SHOULD), **не следует** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **не рекомендуется** (NOT RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе должны интерпретироваться в соответствии с BCP 14 [RFC2119] [RFC8174] тогда и только тогда, когда они выделены шрифтом, как показано здесь.

Значения SID (и рассчитанные по ним детали SID) показаны в примерах лишь для иллюстрации.

Ниже перечислены термины, определённые в [RFC7950].

- action - действие;
- anydata - любые данные;
- anyxml - любые данные XML;
- data node - узел данных;
- data tree - дерево данных;
- datastore - хранилище данных;
- feature - функция, возможность;
- identity - отождествление
- module - модуль;
- notification - уведомление;
- RPC - удаленный вызов процедуры;
- schema node - узел схемы;
- submodule - субмодуль.

В [RFC8040] определён термин

- yang-data extension - расширение yang-data.

В [RFC8791] определён термин

- YANG data structure - структура данных YANG.

Эта спецификация использует также определённые ниже термины.

YANG Schema Item iDentifier (YANG SID или SID) - идентификатор элемента схемы YANG
63-битовое целое число без знака, служащее для идентификации элемента YANG.

delta - приращение
Разница между текущим и эталонным значением YANG SID. Эталон YANG SID определён в каждом контексте, где применяется приращение.

absolute SID - абсолютный SID
Значение YANG SID, не представляемое приращением. Обычно идентификатор называется явно только в позиции, где обычно можно найти приращение.

representation tree - дерево представления
Дерево данных YANG, возможно заключённое в узел схемы, такое как структура данных YANG, уведомление, RPC, действие.

representation node - узел представления
Узел в дереве представления, т. е. узел дерева данных, или представление узла схемы, такое как структура данных YANG, уведомление, RPC, действие.

item - элемент
Узел схемы, отождествление, модуль или свойство, определённые с использованием языка моделирования YANG.

list entry - запись списка
данные, связанные с одной записью в списке (см. параграф 7.8 в [RFC7950]).

container-like instance - контейнероподобный экземпляр
Экземпляр контейнера, структура данных YANG, содержимое уведомления, ввод или вывод RPC или действия (параграф 4.2), запись в списке (параграф 4.4), узел anydata (параграф 4.5).

parent (of a representation node) - родитель узла представления
Узел схемы ближайшего охватывающего узла представления, в котором определён данный узел представления.

3. Свойства кодирования CBOR

Этот документ задаёт правила кодирования CBOR для деревьев данных YANG и их субдеревьев.

Дерево данных YANG может помещаться в представление узла схемы, такое как структура данных YANG, уведомление, RPC или действие, это называется деревом представления. Узлы дерева данных и представление узла схемы, включающего дерево, совместно называют узлами представления (representation node).

Узел представления, такой как контейнер, записи списка, структура данных YANG, уведомление, ввод или вывод RPC или действия, узел anydata, сериализуется с использованием отображения CBOR, где каждый определённый узел схемы кодируется ключом и значением. Эта спецификация поддерживает два типа ключей CBOR - идентификаторы YANG SID (YANG Schema Item iDentifier), определённые в параграфе 3.2, и имена, как указано в параграфе 3.3. Каждый из этих типов ключей кодируется с применением конкретного типа CBOR, что позволяет интерпретировать их в процессе десериализации. Протоколы и механизмы, реализующие эту спецификацию, могут предписывать применение определённого типа ключей или разрешать генератору свободный выбор для каждого ключа.

Для минимизации размера закодированных данных отображение избегает всей необязательной метаинформации, сверх представленной напрямую базовой моделью данных CBOR (раздел 2 в [RFC8949]). Например, теги CBOR применяются лишь для абсолютных SID, узлов данных anyxml и типа данных union, чтобы явно различать типы данных YANG, закодированные с использованием одного базового типа CBOR.

Если иное явно не указано в протоколе или механизме, реализующем эту спецификацию, **нужно** поддерживать в декодерах CBOR, реализующих YANG-CBOR, кодирование неопределённого размера, определённое в параграфе 3.2 [RFC8949]. Это позволяет реализации начинать выдачу массива или отображения до того, как станет известно точное число элементов, а также, возможно, предотвратить неоправданные блокировки и конкуренцию. С другой стороны, это лишает получателя кодированных данных возможности заранее аносировать сведения о размере, когда эти преимущества действительно возникают.

Узлы данных, реализованные с применением массивов, отображения, строк байтов или текста CBOR, могут создаваться пустыми. В этом случае они кодируются с нулевым размером.

При сериализации узлов представления по заданным этой спецификацией правилам как части данных (payload) приложения в эти данные **следует** включать сведения об узлах, позволяющие идентифицировать узлы без хранения состояний, например, связанный с узлом номер SID, приращение SID от другого SID в данных приложения, имя с указанием пространства имён, идентификатор экземпляра.

Примеры в разделе 4 включают корневое отображение CBOR с одной записью, имеющей в качестве ключа имя с указанием пространства имён или SID. Эти корневые отображения CBOR представлены лишь в качестве примеров использования и не являются частью правил кодирования. Обязательным является лишь значение в этом отображении CBOR.

3.1. Диагностическая нотация CBOR

В этом документе двоичное содержимое CBOR представляется с использованием эквивалентной текстовой формы, называемой диагностической нотацией CBOR, как определено в разделе 8 [RFC8949]. Эта нотация служит лишь для документации и не применяется при сериализации данных. Сводка нотации представлена в таблице 1.

Таблица 1. Сводка диагностической нотации CBOR.

Содержимое CBOR	*Тип CBOR*	*Диагностическая нотация*	*Пример*	*Код CBOR*
Целое число без знака	0	Десятичные цифры	123	18 7B
Отрицательное целое число	1	Десятичные цифры со знаком минус (-)	-123	38 7A
Строка байтов	2	Шестнадцатеричное значение в одинарных кавычках или с префиксом h	h'F15C'	42 F15C
Строка текста	3	Строка символов Unicode в двойных кавычках	"txt"	63 747874
Массив	4	Список значений через запятую, заключённых в квадратные скобки	[1, 2]	82 01 02
Отображение	5	Список пар ключ-значение через запятую, заключённых в фигурные скобки	{ 1: 123, 2: 456 }	A2 01187B 021901C8
Логическое значение	7/20	false	false	F4
	7/21	true	true	F5
Null	7/22	null	null	F6
Не выделено	7/23	undefined	undefined	F7

Примечание. Двоичное содержимое CBOR в этой спецификации сопровождается комментариями, обозначенными символами /, как указано в Приложении G.6 [RFC8610].

3.2. Идентификатор элемента схемы YANG

Некоторые из элементов YANG [RFC7950] требуют применения уникального идентификатора. В протоколах настройки конфигурации сети (Network Configuration Protocol или NETCONF) [RFC6241] и RESTCONF [RFC8040] такими идентификаторами служат строки текста. Для обеспечения реализации моделей данных, определённых в YANG, устроствами или сетями с ограничениями нужен более компактный метод идентификации элементов YANG. Такой компактный идентификатор называется идентификатором элемента схемы YANG (YANG Schema Item iDentifier) и является 63-битовым целым числом без знака (0..9223372036854775807 или 0..0x7fffffffffffffff). Ниже указаны элементы, идентифицируемые с помощью YANG SID (или просто SID).

- Отождествления;
- узлы данных,
- RPC и связанные с ними ввод и вывод;
- действия и связанные с ними ввод и вывод;
- структуры данных YANG;
- уведомления и связанная с ними информация;
- модули и функции (feature) YANG.

Отметим, что любое структурирование модулей по субмодулям не заметно для YANG-CBOR, SID не выделяются для имён субмодулей и любые элементы внутри субмодуля фактически получают SID при обработке содержащего их модуля.

Для минимизации размера SID, применяемые как ключи в отображениях CBOR кодируются с использованием приращения целыми числами со знаком, которые складываются с эталонным SID для отображения. Эталонный SID внешнего отображения имеет значение 0, если другой эталонный SID не задан однозначно из среды, где применяется внешнее отображение. Эталонный SID отображения, напрямую встроенного в запись отображения с ключом на основе имени имеет значение 0. Для прочих отображений эталонным SID является значение, рассчитанное для записи отображения, в которую оно встроено напрямую (встраивание может быть опосредованным, если применяется массив, например, в списке YANG). Там, где желательны абсолютные SID (целое число предполагает приращение), они должны указываться абсолютными значениями SID с использованием тега CBOR 47 (как указано в параграфе 4.2.1).

Таким образом, преобразование SID в приращение и обратно не использует состояния и основано исключительно на (де)сериализованных данных, возможно в комбинации с эталонным внешним SID, однозначно заданным средой.

Механизмы и процессы назначения SID элементам YANG и обеспечения их уникальности выходят за рамки этой спецификации. Если нужно применять SID, эта спецификация используется вместе со спецификацией, определяющей управление идентификаторами. Связанный документ [CORE-SID] предназначен для указания способа назначения SID для модулей YANG, поддерживаемых IETF, и рекомендуется для остальных модулей YANG. Данная спецификация разработана для поддержки разных методов назначения в отдельных доменах (областях).

Для предоставления реализациям способа внутренней индикации отсутствия SID значение SID = 0 зарезервировано и не будет выделяться. Оно не применяется при обмене.

3.3. Имя

Эта спецификация поддерживает также представление идентификаторов элементов YANG строками текста, подобно принятому в JSON кодированию данных моделей [RFC7951]. Такой подход может избавить от издержек, связанных с выделением SID. Основным недостатком этого подхода является рост размера закодированных данных.

Идентификаторы элементов YANG по именам **должны** иметь одну из двух форма:

- простой (simple) идентификатор элемента YANG (узла схемы или отождествления);
- заданный с пространством имён (namespace-qualified) идентификатор элемента YANG с префиксом в виде имени модуля, где определён элемент, отделённым двоеточием (:).

Имя модуля задаёт пространство имён для всех элементов YANG, определённых в этом модуле. Если элемент задан в субмодуле, полное имя (с пространством имён) использует имя основного модуля, к которому относится субмодуль.

Синтаксис ABNF [RFC5234] для имени показан на рисунке 1, а создание identifier определено в разделе 14 [RFC7950].

```
name = [identifier ":"] identifier
```

Рисунок 1. Вывод ABNF для простого имени или имени с пространством имён.

Полное (namespace-qualified) имя **должно** применяться для всех элементов верхнего уровня в отображении CBOR, а также во всех случаях, когда пространства имён узла представления и его родителя различаются. В остальных случаях **должны** использоваться простые имена.

Пример определения

```
module example-foomod {
  container top {
    leaf foo {
      type uint8;
    }
  }
}

module example-barmod {
  import example-foomod {
    prefix "foomod";
  }
  augment "/foomod:top" {
    leaf bar {
      type boolean;
    }
  }
}
```

Действительное кодирование CBOR для контейнера top показано ниже.

Диагностическая нотация CBOR

```
{
  "example-foomod:top": {
    "foo": 54,
    "example-barmod:bar": true
  }
}
```

Контейнер top и лист bar, определенные в другом модуле YANG как в родительском контейнере, кодируются с полными именами. Лист foo определён в том же модуле YANG, что и его родительский контейнер, кодируется простым именем.

4. Кодирование узлов представления

Узлы представления, определённые с использованием языка моделирования YANG, кодируются с применением CBOR [RFC8949] по правилам, заданным в этом разделе. Предполагается, что читатель знаком с YANG [RFC7950] и CBOR [RFC8949].

4.1. leaf

Кодирование leaf должно выполняться в соответствии с типом данных листа на основе одного из правил раздела 6.

Приведённые ниже примеры, взятые из [RFC6991] и [RFC7317], демонстрируют кодирование листа hostname с применением SID или имени.

```
typedef domain-name {
  type string {
    pattern
      '((([a-zA-Z0-9_]([a-zA-Z0-9\-_]){0,61})?[a-zA-Z0-9]\.)*'
    + '([a-zA-Z0-9_]([a-zA-Z0-9\-_]){0,61})?[a-zA-Z0-9]\.?)'
    + '|\.';
    length "1..253";
  }
}

leaf hostname {
  type inet:domain-name;
}
```

4.1.1. Использование SID

Как и в последующих примера приращение во внешнем отображении предполагает нулевое значение YANG SID (текущий узел схемы).

Диагностическая нотация CBOR

```
{
  1752 : "myhost.example.com"     / hostname (SID 1752) /
}
```

Представление CBOR

```
A1                                      # map(1)
   19 06D8                              # unsigned(1752)
   72                                   # text(18)
      6D79686F73742E6578616D706C652E636F6D # "myhost.example.com"
```

4.1.2. Использование имени

Диагностическая нотация CBOR

```
{
```

```
    "ietf-system:hostname" : "myhost.example.com"
}
```

Представление CBOR

```
A1                                      # map(1)
   74                                   # text(20)
      696574662D73797374656D3A686F73746E616D65
   72                                   # text(18)
      6D79686F73742E6578616D706C652E636F6D
```

4.2. Контейнеры и другие узлы из дерева данных

Экземпляры контейнеров, структур данных YANG, содержимого уведомлений, ввода и вывода RPC и действий **должны** кодироваться с использованием элемента данных отображения CBOR (базовый тип 5). такое же кодирование применяется для элементов списка (параграф 4.4) и узлов anydata (параграф 4.5). Совместно их называют контейнероподобными экземплярами (container-like instances).

Отображение состоит из пар элементов ключ-значение. Каждому ключу с отображении CBOR назначается идентификатор узла схемы, а значению - значение узла представления в соответствии с типом данных экземпляра.

Эта спецификация поддерживает два типа ключей для отображений CBOR - SID (параграф 3.2) и имя (параграф 3.3).

В приведённых ниже применах, взятых из из [RFC6991] и [RFC7317], показано кодирование представления экземпляра контейнера system-state с использованием SID и имени.

```
typedef date-and-time {
  type string {
    pattern '\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}(\.\d+)?'
          + '(Z|[\+\-]\d{2}:\d{2})';
  }
}

container system-state {
  container clock {
    leaf current-datetime {
      type date-and-time;
    }

    leaf boot-datetime {
      type date-and-time;
    }
  }
}
```

4.2.1. Использование SID

В контексте контейнеров и других узлов дерева данных ключи отображений CBOR во внутренних отображениях можно кодировать с помощью приращений (целое число) или абсоютных SID (с тегом 47). Расчёт приращений показан ниже.

- Для container приращение равно SID текущего узла представления за вычетом SID родительского контейнера.
- Для list приращения равно SID текущего узла представления за вычетом SID родительского списка.
- Для RPC input и RPC output приращения равны SID текущего узла представления за вычетом SID RPC.
- Для action input и action output приращения равны SID текущего узла представления за вычетом SID action.
- Для содержимого уведомления приращение равно SID текущего узла представления минус SID notification.

Диагностическая нотация CBOR

```
{
  1720 : {                          / system-state (SID 1720) /
    1 : {                           / clock  (SID 1721) /
      2 : "2015-10-02T14:47:24Z-05:00", / current-datetime(SID 1723)/
      1 : "2015-09-15T09:12:58Z-05:00"  / boot-datetime (SID 1722) /
    }
  }
}
```

Представление CBOR

```
A1                                      # map(1)
   19 06B8                              # unsigned(1720)
   A1                                   # map(1)
      01                                # unsigned(1)
      A2                                # map(2)
         02                             # unsigned(2)
         78 1A                          # text(26)
            323031352D31302D30325431343A34373A32345A2D30353A3030
         01                             # unsigned(1)
         78 1A                          # text(26)
            323031352D30392D31355430393A31323A35385A2D30353A3030
```

Рисунок 2. Кодирование System State Clock.

4.2.2. Использование имени

Ключи отображений CBOR на основе имён **должны** кодироваться с использованием тектовых строк CBOR (базовый тип 3). Должны применяться полные (namespace-qualified) имена, если пространства имён узла представления и его родителя различаются. В остальных случаях **должны** применяться простые имена. Имена и пространства имён определены в разделе 4 [RFC7951].

Ниже приведён пример кодирования экземпляра узла представления контейнера system с использованием имени.

Диагностическая нотация CBOR

```
{
  "ietf-system:system-state" : {
    "clock" : {
      "current-datetime" : "2015-10-02T14:47:24Z-05:00",
      "boot-datetime" : "2015-09-15T09:12:58Z-05:00"
    }
  }
}
```

Представление CBOR

```
A1                                      # map(1)
   78 18                                # text(24)
      696574662D73797374656D3A73797374656D2D7374617465
   A1                                   # map(1)
      65                                # text(5)
         636C6F636B                     # "clock"
      A2                                # map(2)
         70                             # text(16)
            63757272656E742D6461746574696D65
         78 1A                          # text(26)
            323031352D31302D30325431343A34373A32345A2D30353A3030
         6D                             # text(13)
            626F6F742D6461746574696D65
         78 1A                          # text(26)
            323031352D30392D31355430393A31323A35385A2D30353A3030
```

4.3. leaf-list

Для кодирования leaf-list **должен** использоваться элемент данных массива CBOR (базовый тип 4). Каждая запись массива **должна** кодироваться с использованием одного из правил, заданных в разделе 6.

Приведённые ниже примеры из [RFC6991] и [RFC7317] показывают кодирование экземпляра представления листа-списка search с записями ietf.org и ieee.org.

```
typedef domain-name {
  type string {
    pattern
      '((([a-zA-Z0-9_]([a-zA-Z0-9\-_]){0,61})?[a-zA-Z0-9]\.)*'
    + '([a-zA-Z0-9_]([a-zA-Z0-9\-_]){0,61})?[a-zA-Z0-9]\.?)'
    + '|\.';
    length "1..253";
  }
}

leaf-list search {
  type domain-name;
  ordered-by user;
}
```

4.3.1. Использование SID

Диагностическая нотация CBOR

```
{
  1746 : [ "ietf.org", "ieee.org" ]     / search (SID 1746) /
}
```

Представление CBOR

```
A1                      # map(1)
   19 06D2              # unsigned(1746)
   82                   # array(2)
      68                # text(8)
         696574662E6F7267 # "ietf.org"
      68                # text(8)
         696565652E6F7267 # "ieee.org"
```

4.3.2. Использование имени

Диагностическая нотация CBOR

```
{
  "ietf-system:search" : [ "ietf.org", "ieee.org" ]
}
```

Представление CBOR

```
A1                                      # map(1)
   72                                   # text(18)
      696574662D73797374656D3A736561726368 # "ietf-system:search"
   82                                   # array(2)
      68                                # text(8)
         696574662E6F7267               # "ietf.org"
      68                                # text(8)
         696565652E6F7267               # "ieee.org"
```

4.4. Списки и записи списков

Список или часть списка **должны** кодироваться с использованием элемента данных массива CBOR (базовый тип 4). Каждая запись списка в массиве CBOR кодируется с использованием элемента данных отображения CBOR (базовый тип 5) по правилам кодирования контейнероподобных экземпляров, приведённым в параграфе 4.2.

Важно отметить, что это правило кодирования применимо и к экземплярам узла представления list с 1 элементом.

Пример определения из [RFC7317] показывает кодирование списка server с использованием SID и имени.

```
list server {
  key name;

  leaf name {
    type string;
  }
  choice transport {
    case udp {
      container udp {
        leaf address {
          type host;
          mandatory true;
        }
        leaf port {
          type port-number;
        }
      }
    }
  }
  leaf association-type {
    type enumeration {
      enum server;
      enum peer;
      enum pool;
    }
    default server;
  }
  leaf iburst {
    type boolean;
    default false;
  }
  leaf prefer {
    type boolean;
    default false;
  }
}
```

4.4.1. Использование SID

Правила кодирования для каждой записи list приведены в параграфе 4.2.1.

Диагностическая нотация CBOR

```
{
  1756 : [                    / server (SID 1756) /
    {
      3 : "NRC TIC server",   / name (SID 1759) /
      5 : {                   / udp (SID 1761) /
        1 : "tic.nrc.ca",     / address (SID 1762) /
        2 : 123               / port (SID 1763) /
      },
      1 : 0,                  / association-type (SID 1757) /
      2 : false,              / iburst (SID 1758) /
      4 : true                / prefer (SID 1760) /
    },
    {
      3 : "NRC TAC server",   / name (SID 1759) /
      5 : {                   / udp (SID 1761) /
        1 : "tac.nrc.ca"      / address (SID 1762) /
      }
    }
  ]
}
```

Представление CBOR

```
A1                                      # map(1)
   19 06DC                              # unsigned(1756)
   82                                   # array(2)
      A5                                # map(5)
         03                             # unsigned(3)
         6E                             # text(14)
            4E524320544943207365727665 72 # "NRC TIC server"
         05                             # unsigned(5)
         A2                             # map(2)
            01                          # unsigned(1)
            6A                          # text(10)
               7469632E6E72632E6361     # "tic.nrc.ca"
            02                          # unsigned(2)
```

```
            18 7B                       # unsigned(123)
         01                             # unsigned(1)
         00                             # unsigned(0)
         02                             # unsigned(2)
         F4                             # primitive(20)
         04                             # unsigned(4)
         F5                             # primitive(21)
      A2                                # map(2)
         03                             # unsigned(3)
         6E                             # text(14)
            4E5243205441432073657276657220 # "NRC TAC server"
         05                             # unsigned(5)
         A1                             # map(1)
            01                          # unsigned(1)
            6A                          # text(10)
               7461632E6E72632E6361     # "tac.nrc.ca"
```

4.4.2. Использование имени

Правила кодирования для каждой записи list приведены в параграфе 4.2.2.

Диагностическая нотация CBOR

```
{
  "ietf-system:server" : [
    {
      "name" : "NRC TIC server",
      "udp" : {
        "address" : "tic.nrc.ca",
        "port" : 123
      },
      "association-type" : 0,
      "iburst" : false,
      "prefer" : true
    },
    {
      "name" : "NRC TAC server",
      "udp" : {
        "address" : "tac.nrc.ca"
      }
    }
  ]
}
```

Представление CBOR

```
A1                                  # map(1)
   72                               # text(18)
      696574662D73797374656D3A736572766572
   82                               # array(2)
      A5                            # map(5)
         64                         # text(4)
            6E616D65                # "name"
         6E                         # text(14)
            4E5243205449432073657276657220
         63                         # text(3)
            756470                  # "udp"
         A2                         # map(2)
            67                      # text(7)
               61646472657373       # "address"
            6A                      # text(10)
               7469632E6E72632E6361 # "tic.nrc.ca"
            64                      # text(4)
               706F7274             # "port"
            18 7B                   # unsigned(123)
         70                         # text(16)
            6173736F63696174696F6E2D74797065
         00                         # unsigned(0)
         66                         # text(6)
            696275727374            # "iburst"
         F4                         # primitive(20)
         66                         # text(6)
            707265666572            # "prefer"
         F5                         # primitive(21)
      A2                            # map(2)
         64                         # text(4)
            6E616D65                # "name"
         6E                         # text(14)
            4E5243205441432073657276657220
         63                         # text(3)
            756470                  # "udp"
         A1                         # map(1)
            67                      # text(7)
               61646472657373       # "address"
            6A                      # text(10)
               7461632E6E72632E6361 # "tac.nrc.ca"
```

4.5. anydata

Узлы anydata служат контейнерами для произвольных наборов узлов представления, которые иначе появлялись бы как обычные данные модели YANG. Экземпляр узла представления anydata кодируется с использованием правил, применяемых для контейнеров, т. е. с применением элементов данных отображения CBOR (базовый тип 5) по правилам кодирования контейнероподобных экземпляров, указанным в параграфе 4.2.

Ниже приведён пример, показывающий возможное использование узла anydata. Здесь узел anydata служит для определения узла представления, содержащего уведомление, этот узел может быть частью списка YANG для записи событий.

```
module event-log {
  ...
  anydata last-event;                // SID 60123
}
```

В этом примере предполагается участие приведённого ниже уведомления.

```
module example-port {
  ...

  notification example-port-fault {  // SID 60200
    leaf port-name {                 // SID 60201
      type string;
    }
    leaf port-fault {                // SID 60202
      type string;
    }
  }
}
```

4.5.1. Использование SID

Диагностическая нотация CBOR

```
{
  60123 : {                   / last-event (SID 60123) /
    77 : {                    / example-port-fault (SID 60200) /
      1 : "0/4/21",           / port-name (SID 60201) /
      2 : "Open pin 2"        / port-fault (SID 60202) /
    }
  }
}
```

Представление CBOR

```
A1                              # map(1)
   19 EADB                      # unsigned(60123)
   A1                           # map(1)
      18 4D                     # unsigned(77)
      A2                        # map(2)
         01                     # unsigned(1)
         66                     # text(6)
            302F342F3231        # "0/4/21"
         02                     # unsigned(2)
         6A                     # text(10)
            4F70656E2070696E2032 # "Open pin 2"
```

В некоторых реализациях может быть проще использовать кодирование абсолютных SID (тег 47) для корневого элемента anydata.

Диагностическая нотация CBOR

```
{
  60123 : {                   / last-event (SID 60123) /
    47(60200) : {             / event-port-fault (SID 60200) /
      1 : "0/4/21",           / port-name (SID 60201) /
      2 : "Open pin 2"        / port-fault (SID 60202) /
    }
  }
}
```

4.5.2. Использование имени

Диагностическая нотация CBOR

```
{
  "event-log:last-event" : {
    "example-port:example-port-fault" : {
      "port-name" : "0/4/21",
      "port-fault" : "Open pin 2"
    }
  }
}
```

Представление CBOR

```
A1                                  # map(1)
   74                               # text(20)
      6576656E742D6C6F673A6C6173742D6576656E74
   A1                               # map(1)
      78 1F                         # text(31)
         6578616D706C652D706F72743A
```

```
          6578616D706C652D706F72742D6661756C74
       A2                                  # map(2)
          69                               # text(9)
             706F72742D6E616D65            # "port-name"
          66                               # text(6)
             302F342F3231                  # "0/4/21"
          6A                               # text(10)
             706F72742D6661756C74          # "port-fault"
          6A                               # text(10)
             4F70656E2070696E2032          # "Open pin 2"
```

4.6. anyxml

Узлы представления anyxml служат для сериализации произвольного содержимого CBOR, т. е. значением может быть любой двоичный объект CBOR (xml в названии не вполне корректно, поскольку применяется лишь к YANG-XML [RFC7950]). Значение anyxml **может** содержать элементы данных CBOR, помеченные одним из тегов, указанных в параграфе 9.3. Эти теги **нужно** поддерживать.

В приведённых ниже примерах из [RFC7951] показан действительный экземпляр узла представления anyxml с кодированием CBOR, содержащий массив CBOR с простыми значениями CBOR true, null, true.

```
module bar-module {
  ...
  anyxml bar;      // SID 60000
}
```

4.6.1. Использование SID

Диагностическая нотация CBOR

```
{
  60000 : [true, null, true]   / bar (SID 60000) /
}
```

Представление CBOR

```
A1         # map(1)
   19 EA60 # unsigned(60000)
   83      # array(3)
      F5   # primitive(21)
      F6   # primitive(22)
      F5   # primitive(21)
```

4.6.2. Использование имени

Диагностическая нотация CBOR

```
{
  "bar-module:bar" : [true, null, true]   / bar (SID 60000) /
}
```

Представление CBOR

```
A1                                 # map(1)
   6E                              # text(14)
      6261722D6D6F64756C653A626172 # "bar-module:bar"
   83                              # array(3)
      F5                           # primitive(21)
      F6                           # primitive(22)
      F5                           # primitive(21)
```

5. Кодирование расширения yang-data

Расширение yang-data [RFC8040] служит для определения в YANG структур данных, не предназначенных для реализации как части хранилища. Расширение yang-data будет задавать контейнер, который должен кодироваться по правилам для узлов дерева данных, указанным в параграфе 4.2. Как и контейнеры YANG, расширение yang-data можно кодировать с использованием SID или имени.

Пример определения из Приложения A к [CORE-COMI]

```
module ietf-coreconf {
  ...

  import ietf-restconf {
    prefix rc;
  }

  rc:yang-data yang-errors {
    container error {
      leaf error-tag {
        type identityref {
          base error-tag;
        }
      }
      leaf error-app-tag {
        type identityref {
          base error-app-tag;
        }
      }
      leaf error-data-node {
        type instance-identifier;
      }
```

```
        leaf error-message {
          type string;
        }
      }
    }
  }
```

5.1. Использование SID

Расширения yang-data, закодированные с использованием SID, передаются в отображении CBOR с одной парой элементов. Для ключа устанавливается значение SID, выделенное контейнеру расширения yang-data, а в значение помещается CBOR-кодирование этого контейнера, как задано в параграфе 4.2.

Приведённый ниже пример сериализации экземпляра yang-errors расширения yang-data, как определено в [CORE-COMI], с использованием SID в соответствии с параграфом 3.2.

Диагностическая нотация CBOR

```
{
  1024 : {                      / error  (SID 1024) /
    4 : 1011,                   / error-tag (SID 1028) /
                                / = invalid-value (SID 1011) /
    1 : 1018,                   / error-app-tag (SID 1025) /
                                / = not-in-range (SID 1018) /
    2 : 1740,                   / error-data-node (SID 1026) /
                                / = timezone-utc-offset (SID 1740) /
    3 : "Maximum exceeded"      / error-message (SID 1027) /
  }
}
```

Представление CBOR

```
A1                                      # map(1)
   19 0400                              # unsigned(1024)
   A4                                   # map(4)
      04                                # unsigned(4)
      19 03F3                           # unsigned(1011)
      01                                # unsigned(1)
      19 03FA                           # unsigned(1018)
      02                                # unsigned(2)
      19 06CC                           # unsigned(1740)
      03                                # unsigned(3)
      70                                # text(16)
         4D6178696D756D206578636565646564 # "Maximum exceeded"
```

5.2. Использование имени

Расширение yang-data закодированное с использованием имени передаётся в отображении CBOR с одной парой элементов. Для ключа устанавливается полное (namespace-qualified) имя контейнера расширения yang-data, а значением служит CBOR-кодирование этого контейнера, как указано в параграфе 4.2.

Ниже приведён пример сериализации экземпляра yang-errors расширения yang-data, определённого в [CORE-COMI], с использованием имени, как указано в параграфе 3.3.

Диагностическая нотация CBOR

```
{
  "ietf-coreconf:error" : {
    "error-tag" : "invalid-value",
    "error-app-tag" : "not-in-range",
    "error-data-node" : "timezone-utc-offset",
    "error-message" : "Maximum exceeded"
  }
}
```

Представление CBOR

```
A1                                          # map(1)
   73                                       # text(19)
      696574662D636F7265636F6E663A6572726F72 # "ietf-coreconf:error"
   A4                                       # map(4)
      69                                    # text(9)
         6572726F722D746167                 # "error-tag"
      6D                                    # text(13)
         696E76616C69642D76616C7565         # "invalid-value"
      6D                                    # text(13)
         6572726F722D6170702D746167         # "error-app-tag"
      6C                                    # text(12)
         6E6F742D696E2D72616E6765           # "not-in-range"
      6F                                    # text(15)
         6572726F722D646174612D6E6F6465     # "error-data-node"
      73                                    # text(19)
         74696D657A6F6E652D7574632D6F6666736574
                                            # "timezone-utc-offset"
      6D                                    # text(13)
         6572726F722D6D657373616765         # "error-message"
      70                                    # text(16)
         4D6178696D756D206578636565646564   # "Maximum exceeded"
```

6. Представление типов данных YANG в CBOR

Кодирование CBOR для экземпляра узла представления leaf или leaf-list зависит от встроенного типа этого узла представления. Ниже определяется кодирование CBOR для каждого встроенного типа, поддерживаемого в YANG (см. параграф 4.2.4 в [RFC7950]). В каждом параграфе приведён пример значения, выделенного экземпляру узла представления обсуждаемого встроенного типа.

6.1. Целочисленные типы без знака

Листья типов uint8, uint16, uint32, uint64 **должны** кодироваться с использованием беззнакового целочисленного элемента данных CBOR (базовый тип 0).

Пример из [RFC8344] показывает кодирование экземпляра узла представления mtu со значением 1280 байтов.

```
leaf mtu {
  type uint16 {
    range "68..max";
  }
}
```

Диагностическая нотация CBOR: 1280

Представление CBOR: 19 0500

6.2. Целочисленные типы

Листья типов int8, int16, int32, int64 **должны** кодироваться с использованием беззнакового целочисленного элемента данных CBOR (базовый тип 0) или отрицательного целого числа CBOR (базовый тип 1), в зависимости от фактического значения.

Пример из [RFC7317] показывает кодирование экземпляра узла представления timezone-utc-offset со значением -300 минут.

Пример определения из [RFC7317]

```
leaf timezone-utc-offset {
  type int16 {
    range "-1500 .. 1500";
  }
}
```

Диагностическая нотация CBOR: -300

Представление CBOR: 39 012B

6.3. decimal64

Листья типа decimal64 **должны** кодироваться с использованием десятичной дроби, как указано в параграфе 3.4.4 [RFC8949].

Пример из [RFC7317] показывает кодирование экземпляра узла представления my-decimal со значением 2,57.

```
leaf my-decimal {
  type decimal64 {
    fraction-digits 2;
    range "1 .. 3.14 | 10 | 20..max";
  }
}
```

Диагностическая нотация: CBOR 4([-2, 257])

Представление CBOR: C4 82 21 19 0101

6.4. string

Листья типа string **должны** кодироваться с использованием текстовой строки CBOR (базовый тип 3).

Пример из [RFC8343] показывает кодирование экземпляра узла представления name со значением eth0.

```
leaf name {
  type string;
}
```

Диагностическая нотация CBOR: "eth0"

Представление CBOR: 64 65746830

6.5. boolean

Листья типа boolean **должны** кодироваться с использованием простого значения CBOR true (базовый тип 7, дополнительное значение 21) или false (базовый тип 7, дополнительное значение 20).

Пример из [RFC7317] показывает кодирование экземпляра узла представления enabled со значением true.

```
leaf enabled {
  type boolean;
}
```

Диагностическая нотация CBOR: true

Представление CBOR: F5

6.6. enumeration

Листья типа enumeration **должны** кодироваться с использованием беззнакового целочисленного элемента данных CBOR (базовый тип 0) или отрицательного целого числа CBOR (базовый тип 1), в зависимости от фактического значения, или, в качестве исключения, как строка текста с тегом (см. ниже). Перечисляемые значения выделяются явно с использованием оператора YANG value или автоматически на основе алгоритма, описанного в параграфе 9.6.4.2 [RFC7950].

Пример из [RFC7317] показывает кодирование экземпляра узла представления oper-status со значением testing.

```
leaf oper-status {
  type enumeration {
    enum up { value 1; }
    enum down { value 2; }
    enum testing { value 3; }
    enum unknown { value 4; }
    enum dormant { value 5; }
    enum not-present { value 6; }
    enum lower-layer-down { value 7; }
  }
}
```

Диагностическая нотация CBOR: 3

Представление CBOR: 03

Значения типа enumeration, заданные в типе union **должны** кодироваться с использованием строки текста CBOR (базовый тип 3) и **должны** содержать одно из имён, назначенных оператором enum в YANG (см. параграф 6.12). Кодирование **должно** использовать тег перечисления CBOR, как указано в параграфе 9.3.

Пример определения из [RFC7950]

```
type union {
  type int32;
  type enumeration {
    enum unbounded;
  }
}
```

Диагностическая нотация CBOR: 44("unbounded")

Представление CBOR: D8 2C 69 756E626F756E646564

6.7. bits

С учётом того, что биты назначаются явно с помощью оператора YANG position или автоматически на основе алгоритма, заданного в параграфе 9.7.4.2 [RFC7950], каждый элемент типа bits можно рассматривать как набор битовых позиций (или смещений от позиции 0), имеющих значение 1 (бит установлен) или 0 (бит сброшен).

Листья типа bits **должны** кодироваться с использованием массива CBOR (базовый тип 4) или строки байтов (базовый тип 2), а в исключительных случаях - строки текста с тегом (см. ниже). При использовании массива CBOR каждый элемент является (1) положительным целым числом (базовый тип 0, дополнительное значение 0 не разрешено), которое служит для расчёта смещения следующей за ним строки байтов, или (2) строкой байтов (базовый тип 2) с информацией об установленных и сброшенных битах. Начальное смещение имеет значение 0 и каждое целое число без знака меняет смещение следующей за ним строки байтов на целое число, умноженное на 8. Например, если смещение бита равно 0 и имеется целое число 5, первый байт следующей строки байтов будет представлять битовые позиции от 40 до 47, включительно. Если строка байтов имеет второй байт, он будет содержать сведения о битах 48 - 55 и т. д. Внутри каждого байта биты назначаются от младшего к старшему. После строки байтов смещение меняется на число байтов в строке, умноженное на 8. Байты без установленных битов (0) в конце строки байтов не создаются. Если это происходит в конце массива, такие байты просто опускаются. Если же это возникает в конце строки байтов, предшествующей целому числу, байты с нулями удаляются и это целое число увеличивается на количество удалённых байтов с нулями.

В примере представлено кодирование экземпляра узла представления alarm-state с установленными флагами critical (позиция 2), warning (позиция 8) и indeterminate (позиция 128).

```
typedef alarm-state {
  type bits {
    bit unknown;
    bit under-repair;
    bit critical;
    bit major;
    bit minor;
    bit warning {
      position 8;
    }
    bit indeterminate {
      position 128;
    }
  }
}

leaf alarm-state {
  type alarm-state;
}
```

Диагностическая нотация CBOR: [h'0401', 14, h'01']

Представление CBOR: 83 42 0401 0E 41 01

Во многих случаях массив будет содержать лишь 1 элемент - строку из нескольких байтов. В таком случае **требуется** опустить элемент массива и включать только строку (массив) байтов. В качестве примера рассмотрим приведённое выше определение YANG с кодированием лишь флагов under-repair и critical. Результат показан ниже.

Диагностическая нотация CBOR: h'06'

Представление CBOR: 41 06

Элементы массива **должны** быть строками байтов без нулевых байтов в конце или положительными целыми числами, которые **должны** чередоваться со строками байтов, т. е. два целых числа или две строки байтов подряд являются ошибкой. Массив с одной строкой байтов **должен** кодироваться просто как строка байтов. Массив с одним положительным целым числом является ошибкой. Отметим, что получатель может обрабатывать нулевые байты в конце по обычным правилам без каких-либо проблем, поэтому реализация **может** просто воспринимать их.

Значения типа bits, определённые в типе union, **должны** кодироваться с использованием строки текста CBOR (базовый тип 3) и **должны** содержать последовательность разделённых пробелами имён bits, которые установлены (см. параграф 6.12). Кодирование **должно** использовать тег CBOR, как указано в параграфе 9.3.

В примере показано кодирование экземпляра узла представления alarm-state, заданного с использованием типа union, где установлены флаги under-repair и critical.

```
leaf alarm-state-2 {
  type union {
    type alarm-state;
    type bits {
      bit extra-flag;
    }
  }
}
```

Диагностическая нотация CBOR: 43("under-repair critical")

Представление CBOR: D8 2B 75 756E6465722D726570616972206372697469963616C

6.8. binary

Листья типа binary **должны** кодироваться с использованием строки байтов CBOR (базовый тип 2).

Пример показывает кодирование экземпляра узла представления aes128-key со значением 0x1f1ce6a3f42660d888d92a4d8030476e.

```
leaf aes128-key {
  type binary {
    length 16;
  }
}
```

Диагностическая нотация CBOR: h'1F1CE6A3F42660D888D92A4D8030476E'

Представление CBOR 50: 1F1CE6A3F42660D888D92A4D8030476E

6.9. leafref

Листья типа leafref **должны** кодироваться по правилам узла представления, указанного оператором YANG path.

Приведённый пример из [RFC8343] показывает кодирование экземпляра узла представления interface-state-ref со значением eth1.

```
typedef interface-state-ref {
  type leafref {
    path "/interfaces-state/interface/name";
  }
}

container interfaces-state {
  list interface {
    key "name";
    leaf name {
      type string;
    }
    leaf-list higher-layer-if {
      type interface-state-ref;
    }
  }
}
```

Диагностическая нотация CBOR: "eth1"

Представление CBOR: 64 65746831

6.10. identityref

Спецификация поддерживает два подхода к кодированию identityref с применением YANG SID (параграф 3.2) или имени (параграф 6.8 в [RFC7951]). Исключение, требующее применять тег, описано в параграфе 6.12.

6.10.1. SID как identityref

При реализации узла представления типа identityref с применением SID он **должен** кодироваться целым числом без знака CBOR (базовый тип 0). Отметим, что механизм приращений для SID в качестве identityref не применяется, поскольку они не используются как ключи отображения CBOR.

Пример из [RFC7317] показывает кодирование экземпляра узла представления type со значением iana-if-type:ethernetCsmacd (SID 1880).

```
identity interface-type {
}

identity iana-interface-type {
  base interface-type;
}

identity ethernetCsmacd {
  base iana-interface-type;
}

leaf type {
  type identityref {
    base interface-type;
  }
}
```

Диагностическая нотация CBOR: 1880

Представление CBOR: 19 0758

6.10.2. Имя как identityref

Можно кодировать identityref с использованием имени, как указано в параграфе 3.3. В этом случае для кодирования identityref **должна** использоваться строка текста CBOR (базовый тип 3). Если отождествление задано не в том модуле, где содержится узел данных identityref, **должно** указываться полное (namespace-qualified) имя, в остальных случаях разрешены обе формы имён (см. параграф 3.3).

В примере показано кодирование отождествления iana-if-type:ethernetCsmacd с применением полного имени (см. параграф 6.10.1).

Диагностическая нотация CBOR: "iana-if-type:ethernetCsmacd"

Представление CBOR: 78 1B 69616E612D69662D747970653A65746865726E657443736D616364

6.11. empty

Листья типа empty **должны** кодироваться с использованием значения CBOR null (базовый тип 7, дополнительное значение 22).

Пример из [RFC8344] показывает кодирование экземпляра узла представления is-router, когда он имеется.

```
leaf is-router {
  type empty;
}
```

Диагностическая нотация CBOR: null

Представление CBOR: F6

6.12. union

Листья типа union **должны** кодироваться по правилам, связанным с одним из перечисленных ниже типов. При использовании в union указанные ниже типы данных YANG применяются с тегом CBOR для исключения путаницы между разными типами YANG, кодируемыми с использованием одного базового типа CBOR.

- bits;
- enumeration;
- identityref;
- instance-identifier.

Значения тегов CBOR указаны в параграфе 9.3.

Как отмечено в параграфах 6.6 и 6.7, типы enumeration и bits кодируются с помощью текстовых строк CBOR (базовый тип 3) при определении внутри типа union. Это повышает сложность, но необходимо из-за особенностей модели данных YANG для union - обходной путь обеспечивает совместимость с кодированием перекрывающихся объединений (union) в XML и JSON (см. параграф 9.12 в [RFC7950].)

Пример из [RFC6991] показывает кодирование экземпляра узла представления ip-address для значения 2001:db8:a0b:12f0::1.

```
typedef ipv4-address {
  type string {
    pattern
      '(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}'
    +  '([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])'
    + '(%[\p{N}\p{L}]+)?';
  }
}

typedef ipv6-address {
  type string {
    pattern '((:|[0-9a-fA-F]{0,4}):)([0-9a-fA-F]{0,4}:){0,5}'
          + '((([0-9a-fA-F]{0,4}:)?(:|[0-9a-fA-F]{0,4}))|'
```

```
            + '(((25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9])\.){3}'
            + '(25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9])))'
            + '(%[\p{N}\p{L}]+)?';
      pattern '(([^:]+:){6}(([^:]+:[^:]+)|(.*\..*)))|'
            + '((([^:]+:)*[^:]+)?::(([^:]+:)*[^:]+)?)'
            + '(%.+)?';
    }
  }

  typedef ip-address {
    type union {
      type ipv4-address;
      type ipv6-address;
    }
  }

  leaf address {
    type ip-address;
  }
```

Диагностическая нотация CBOR: "2001:db8:a0b:12f0::1"

Представление CBOR: 74 323030313A6462383A6130623A313266303A3A31

6.13. instance-identifier

Эта спецификация поддерживает два подхода к кодированию instance-identifier на основе YANG SID (параграф 3.2) и имён (параграф 3.3). Исключение, требующее применять тег, описано в параграфе 6.12.

6.13.1. SID как instance-identifier

SID однозначно указывает узел схемы. В случае одного экземпляра узла схемы, т. е. при его определении в корне модуля или субмодуля YANG или определении узлов схемы в контейнере, SID достаточно для идентификации экземпляра (узла представления). Отметим, что механизм приращений для SID не применяется (см. параграф 6.10.1.)

В случае узла представления, являющегося записью списка YANG значение SID комбинируется с ключами списка для идентификации каждого экземпляра в списке YANG.

В случае одного экземпляра кодирование instance-identifier **должно** использовать целое число без знака CBOR (базовый тип 0) со значением SID целевого узла схемы.

В списке YANG для кодирования instance-identifier **должен** применяться массив CBOR (базовый тип 4), как указано ниже:

- первый элемент **должен** кодироваться как целое число без знака (базовый тип 0) со значением SID целевого узла схемы;
- следующие элементы **должны** содержать значение ключа, требуемого для идентификации целевого узла схемы. Эти ключи **должны** быть упорядочены, как указано в операторе YANG key, начиная со списка верхнего уровня и продолжая подчинёнными списками.

Примеры в этом параграфе предполагают определение узла схемы типа instance-identifier. Пример определения из [RFC7950]

```
  container system {
    ...
    leaf reporting-entity {
      type instance-identifier;
    }
```

Первый пример

Пример определения из [RFC7317] показывает кодирование экземпляра узла представления reporting-entity со значением /system/contact (SID 1741).

```
  container system {

    leaf contact {
      type string;
    }

    leaf hostname {
      type inet:domain-name;
    }
  }
```

Диагностическая нотация CBOR: 1741

Представление CBOR: 19 06CD

Второй пример

Этот пример предназначен для демонстрации идентификации записи узла представления в списке YANG с использованием изменённой версии модуля YANG из [RFC7317] (добавление страны к листься и уполномоченным ключам). Пример иллюстрирует кодирование значения reporting-entity. Указывающего экземпляр списка /system/authentication/user/authorized-key/key-data (предполагается SID 1734) для имени пользователя bob и уполномоченного ключа с именем admin и страной france.

```
  list user {
    key name;

    leaf name {
```

```
      type string;
    }

    leaf password {
      type ianach:crypt-hash;
    }

    list authorized-key {
      key "name country";

      leaf country {
        type string;
      }

      leaf name {
        type string;
      }

      leaf algorithm {
        type string;
      }

      leaf key-data {
        type binary;
      }
    }
  }
```

Диагностическая нотация CBOR: [1734, "bob", "admin", "france"]

Представление CBOR

```
84                  # array(4)
   19 06C6          # unsigned(1734)
   63               # text(3)
      626F62        # "bob"
   65               # text(5)
      61646D696E    # "admin"
   66               # text(6)
      6672616E6365 # "france"
```

Третий пример

Приведённый ниже пример показывает кодирование значения reporting-entity, указывающего экземпляр списка /system/authentication/user (SID 1730), соответствующий имени пользователя jack.

Диагностическая нотация CBOR: [1730, "jack"]

Представление CBOR

```
82             # array(2)
   19 06C2     # unsigned(1730)
   64          # text(4)
      6A61636B # "jack"
```

6.13.2. Имена как instance-identifier

Значение instance-identifier кодируется как строка текста, аналогично лексическому представлению в XML (см. параграф 9.13.2 в [RFC7950]). Однако кодирование пространства имён в instance-identifier следует правилам параграфа 3.3:

- самое левое (верхнего уровня) имя узла данных всегда имеет полную форму (namespace-qualified);
- последующие имена узлов имеют полную форму, если узел определён не в родительском модуле, иначе применяется простая форма (это справедливо и для имён узлов в предикатах).

Например, /ietf-interfaces:interfaces/interface[name='eth0']/ietf-ip:ipv4/ip является действительным значением instance-identifier, поскольку узлы данных interfaces, interface, name определены в модуле ietf-interfaces, а ipv4 и ip - в ietf-ip.

Полученное в результате значение XML Path Language (XPath) **должно** кодироваться с использованием строки текста CBOR (базовый тип 3).

Ниже приведены примеры, описанные в параграфе 6.13.1.

Первый пример

Диагностическая нотация CBOR: "/ietf-system:system/contact"

Представление CBOR

```
78 1B 2F696574662D73797374656D3A73797374656D2F636F6E74616374
```

Второй пример

Диагностическая нотация CBOR (the line break is inserted for exposition only)

```
"/ietf-system:system/authentication/user[name='bob']/
authorized-key[name='admin'][country='france']/key-data"
```

Представление CBOR

```
78 6B
   2F696574662D73797374656D3A73797374656D2F61757468656E74696361
   74696F6E2F757365725B6E616D653D27626F62275D2F617574686F72697A
   65642D6B65795B6E616D653D2761646D696E275D5B636F756E7472793D27
   6672616E6365275D2F6B65792D64617461
```

Третий пример

Диагностическая нотация CBOR

```
"/ietf-system:system/authentication/user[name='jack']"
```

Представление CBOR

```
78 34                                   # text(52)
   2F696574662D73797374656D3A73797374656D2F61757468656E74696361
   74696F6E2F757365725B6E616D653D276A61636B275D
```

7. Content-Type

Эта спецификация определяет тип носителя application/yang-data+cbor, который можно использовать без параметров или с параметром id, содержащим имя (name) или SID (sid). Этот тип носителя представляет документ YANG-CBOR, содержащий дерево представления. При наличии параметра id в зависимости от его значения каждый узел представления указывается связанным с ним полным именем (namespace-qualified), как указано в параграфе 3.3 (id=name) или YANG SID (представленным, например, ключом отображения CBOR как приращение SID или с помощью тега 47), как указано в параграфе 3.2 (id=sid). При отсутствии параметра id могут применяться обе формы.

Форматом представления application/yang-data+cbor является отображение CBOR, сопоставляющее имена и/или SID (как указано выше) со значениями экземпляров (по правилам раздела 4).

В настоящее время не пердполагается использование для параметра id значений, отличающихся от имени и SID, или отсутствие параметра. Если такое расширение произойдёт, предполагается, что новые значения будут иметь форму [a-z][a-z0-9]*(-[a-z0-9]+)*.

Таким образом, документ задаёт 3 content-type, предназначенных для разных классов приложений.

- application/yang-data+cbor, id=sid - для приложений, которым требуется экономить пространство кодирования и обработку строк текста (например, приложения на узлах с ограничениями [RFC7228] или с особыми требованиями к производительности).
- application/yang-data+cbor, id=name - для приложений, которые не хотят поддерживать SID и располагают достаточными ресурсами для работы со строками текста (например, приложения, желающие напрямую заменить application/yang.data+json более эффективным представлением без внесения других изменений).
- application/yang-data+cbor - для комплексных приложений, которые могут получить преимущество от повышенной эффективности идентификаторов SID, сохраняя интеграцию с базами модулей YANG, пока для них не заданы отображения SID.

Все три content-type основаны на одних механизмах представления, части которых в первом и втором случае просто не используются.

Использование одного из трёх content-type в транспортном протоколе выходит за рамки спецификации. В последнем абзаце параграфа 5.2 [RFC8040] рассматривается индикация и запрос использования конкретных content-type в RESTCONF. Похожие механизмы доступны в протоколе ограниченных приложений (Constrained Application Protocol или CoAP) [RFC7252] на основе опций Content-Format и Accept. В [CORE-COMI] показано, как можно использовать Content-Format для индикации в случае id=sid.

8. Вопросы безопасности

Применимы соображения безопасности, отмеченные в [RFC8949] и [RFC7950].

Этот документ определяет дополнительное кодирование данных, смоделированных на языке YANG. Кодирование само по себе не создаёт новых проблем безопасности в дополнение к отмеченным для конкретного проткола и контекста применения.

Для снижения рисков приложениям на приёмной стороне **следует** отвергать все сообщения, не соответствующие правилам этого документа, возвращая отправителю подходящее сообщение об ошибке. Например, при использовании параметра id для типа носителя важно корректно отклонять идентификаторы иного типа, чтобы избежать ситуаций, когда другие реализации по-иному интерпретируют такое содержимое.

При использовании SID интерпретация кодированных данных зависит не только от наличия корректных модулей YANG, но и от правильного отображения SID. Поэтому поддержка и развитие сведений об отображении требует такой же осторожности, как и управление модулями YANG. Процедуры [CORE-SID] учитывают это обстоятельство.

9. Взаимодействие с IANA

9.1. Реестр Media Types

Агентство IANA добавило указанный в таблице 2 тип носителя в реестр Media Types [IANA.media-types].

Таблица 2. Реестр Media Types.

Имя	***Шаблон***	***Документ***
yang-data+cbor	application/yang-data+cbor	RFC 9254

Type name: application
Subtype name: yang-data+cbor
Required parameters: N/A
Optional parameters: id (раздел 7 в RFC 9254)
Encoding considerations: binary (CBOR)
Security considerations: раздел 8 в RFC 9254
Interoperability considerations: N/A
Published specification: RFC 9254

Applications that use this media type: Приложения, которым нужно компактное и эффективное представление данных моделей YANG
Fragment identifier considerations: Синтаксис и семантика идентификатров фрагментов для application/yang-data+cbor совпадают с заданными для application/cbor (на момент публикации этого документа синтаксис идентификации фрагментов для application/cbor не был определен)
Additional information:
Magic number(s): N/A
File extension(s): N/A
Macintosh file type code(s): N/A
Person & email address to contact for further information: почтовая конференция CORE WG (core@ietf.org) или IETF Applications and Real-Time Area (art@ietf.org)
Intended usage: COMMON
Restrictions on usage: N/A
Author: CoRE WG
Change controller: IETF

9.2. Реестр CoAP Content-Formats

Агентство IANA добавило указанные в таблице 3 значения Content-Formats в субреестр CoAP Content-Formats реестра Constrained RESTful Environments (CoRE) Parameters [IANA.core-parameters]. Применяется процедура Expert Review для диапазона 0 - 255 и IETF Review - для 256 - 9999.

Таблица 3. Реестр CoAP Content-Format.

Тип носителя	*Кодирование*	*Идентификатор*	*Документ*
application/yang-data+cbor	-	340	RFC 9254
application/yang-data+cbor; id=name	-	341	RFC 9254
application/yang-data+cbor; id=sid	-	140	RFC 9254

9.3. Реестр CBOR Tags

Агентство IANA выделило номера тегов CBOR в реестре CBOR Tags [IANA.cbor-tags] заданнов в параграфе 9.2 [RFC8949].

Таблица 4. Реестр CBOR Tags

Тег	*Элемент данных*	*Семантика*	*Документ*
43	Строка текста	Тип YANG bits, параграф 6.7.	RFC 9254
44	Строка текста	Тип YANG enumeration, параграф 6.6.	RFC 9254
45	Целое число без знака или строка текста	Тип YANG identityref, параграф 6.10.	RFC 9254
46	Целое число без знака, строка текста или массив	Тип YANG instance-identifier, параграф 6.13.	RFC 9254
47	Целое число без знака	Тип YANG Schema Item iDentifier (SID), параграф 3.2.	RFC 9254

10. Литература

10.1. Нормативные документы

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

[IANA.core-parameters] IANA, "Constrained RESTful Environments (CoRE) Parameters", <https://www.iana.org/assignments/core-parameters/>.

[IANA.media-types] IANA, "Media Types", <https://www.iana.org/assignments/media-types/>.

[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>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[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>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[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>.

[RFC8791] Bierman, A., Björklund, M., and K. Watsen, "YANG Data Structure Extensions", RFC 8791, DOI 10.17487/RFC8791, June 2020, <https://www.rfc-editor.org/info/rfc8791>.

[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>.

10.2. Дополнительная литература

[CORE-COMI] Veillette, M., Ed., van der Stok, P., Ed., Pelov, A., Bierman, A., and I. Petrov, Ed., "CoAP Management Interface (CORECONF)", Work in Progress, Internet-Draft, draft-ietf-core-comi-11, 17 January 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-core-comi-11>.

[CORE-SID] Veillette, M., Ed., Pelov, A., Ed., Petrov, I., Ed., Bormann, C., and M. Richardson, "YANG Schema Item iDentifier (YANG SID)", Work in Progress, Internet-Draft, draft-ietf-core-sid-18, 18 November 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-core-sid-18>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

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

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <https://www.rfc-editor.org/info/rfc7317>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8344] Bjorklund, M., "A YANG Data Model for IP Management", RFC 8344, DOI 10.17487/RFC8344, March 2018, <https://www.rfc-editor.org/info/rfc8344>.

Благодарности

Этот документ в значительной мере вдохновлен обширной работой Andy Bierman и Peter van der Stok над [CORE-COMI]. Докумен [RFC7951] также внёс важный вклад в эту работу. Спасибо авторам и участникам создания этих документов.

Авторы хотели бы также поблагодарить Ladislav Lhotka и Jürgen Schönwälder, а также «куратора» документа Marco Tiloca за обзор, отзывы и комментации. Обширные комментарии в процессе рецензирования IESG помогли улучшить документ и авторы хотели бы особо отметить отзывы и рекомендации от руководителя направления (AD) Francesca Palombini, а также существенные улучшения, предложенные членами IESG Benjamin Kaduk и Rob Wilton.

Адреса авторов

Michel Veillette (editor)
Trilliant Networks Inc.
610 Rue du Luxembourg
Granby Quebec J2J 2V2
Canada
Email: michel.veillette@trilliantinc.com

Ivaylo Petrov (editor)
Google Switzerland GmbH
Brandschenkestrasse 110
CH-8002 Zurich
Switzerland
Email: ivaylopetrov@google.com

Alexander Pelov
Acklio
1137A avenue des Champs Blancs
35510 Cesson-Sevigne Cedex
France
Email: a@ackl.io

Carsten Bormann
Universität Bremen TZI
Postfach 330440
D-28359 Bremen
Germany
Phone: +49-421-218-63921
Email: cabo@tzi.org

Michael Richardson
Sandelman Software Works
Canada
Email: mcr+ietf@sandelman.ca

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru