Internet Engineering Task Force (IETF)
Request for Comments: 8528
Category: Standards Track
ISSN: 2070-1721

M. Bjorklund
Tail-f Systems
L. Lhotka
CZ.NIC

March 2019

YANG Schema Mount

Монтирование схемы YANG

Аннотация

Этот документ определяет механизм добавления элементов деревьев схемы, определённых набором модулей YANG, в точку монтирования дерева схемы в другом модуле YANG .

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 RFC 7841.

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке https://www.rfc-editor.org/info/rfc8528.

Оглавление

1. Введение

Модульность и расширяемость являются одними из ведущих принципов разработки языка моделирования данных YANG. В результате модуль YANG можно комбинировать с разными наборами других модулей для формирования модели данных, приспособленной к требованиям конкретного варианта применения. От разработчиков серверов требуется лишь задать все модули YANG, составляющие модель данных (вместе с их выпусками и другими необязательными параметрами), в данных библиотеки YANG ([RFC7895], [RFC8525] и параграф 5.6.4 в [RFC7950]),

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

реализованной сервером. Такие модули YANG появляются в модели данных «бок о бок», т. е. узлы верхнего уровня каждого модуля (при наличии) будут узлами верхнего уровня модели данных в целом.

В YANG имеется два механизма лобавления иерархии схемы, определённой в другом месте, к содержимому внутреннего узла дерева схемы. Эти механизмы реализуются указанными ниже операторами YANG.

- Оператор uses явно встраивает содержимое группировки, определённой в том же или ином модуле (см. параграф 4.2.6 в [RFC7950]).
- Оператор augment явно добавляет содержимое к целевому узлу, определённому в том же или ином модуле (см. параграф 4.2.8 в [RFC7950]).

В обоих случаях модуль YANG с оператором uses или augment явно определяет точное место в дереве схемы, куда помещаются новые узлы.

В некоторых случаях этих механизмов недостаточно - иногда нужно добавить имеющийся модуль (набор модулей) в модель данных, начиная с места, отличающегося от корня. Например, имеются модули YANG, такие как ietf-interfaces [RFC8343], для использования в модели данных физического устройства. Предположим, что нужна модель устройства, поддерживающего множество логических устройств [RFC8530], каждое из которых имеет свой экземпляр ietf-interfaces и, возможно, других модулей. В то же время нужна возможность управления всеми этими логическими устройствами с одного главного устройства. Для этого нужно дерево схемы, показанное ниже.

```
+--rw interfaces
|  +--rw interface* [name]
|     ...
+--rw logical-network-element* [name]
   +--rw name
   |   ...
   +--rw interfaces
     +--rw interface* [name]
        ...
```

При использовании оператора uses полное дерево схемы ietf-interfaces будет помещено в группировку, а затем эта группировка должна будет использоваться на верхнем уровне (для главного устройства), а также в списке logical-network-element (для логических устройств). Здесь возникает несколько недостатков.

- Не поддерживается расширяемость, поскольку при каждом добавлении в модель логического устройства нового модуля YANG придётся добавлять в эту модель другой оператор uses, извлекающий содержимое нового модуля.
- Абсолютные ссылки на узлы группировки могут не работать, если группировка применяется в разных местах.
- Узлы, заданные внутри группировки, относятся к пространству имён, в котором группировка применяется, что затрудняет или даже делает невозможными ссылки на такие узлы из других модулей.
- Производителям сложно добавлять фирменные (proprietary) модули, когда операторы uses определены в стандартном модуле.

При использовании оператора augment модуль ietf-interfaces будет дополнять все узлы списка logical-network-element, в то же время определяя все узлы списка на верхнем уровне. В результате одна и та же иерархия узлов будет задана дважды, что явно не способствует расширяемости.

В этом документе задан новый механизм, названный монтированием схемы (schema mount), который позволяет смонтировать одну модель данных, состоящую из любого числа модулей YANG, в заданном месте другой (родительской) схемы. В отличие от рассмотренных выше подходов с операторами uses и augment, монтируемые модули не требуют специальной подготовки, поэтому имеющиеся модули (такие как ietf-interfaces) можно монтировать без каких-либо изменений.

Основная идея монтирования схемы состоит в том, что в родительской схеме узел данных помечается как точка монтирования, а затем определяется полная модель данных для присоединения к точке монтирования так, что помеченный узел данных фактически становится корневым узлом примонтированной модели данных. В принципе, монтируемую схему можно задать на трёх разных этапах жизненного цикла модели данных.

1. При разработке монтируемая схема задаётся вместе с точкой монтирования в родительском модуле YANG. В этом случае монтируемая схема будет одинакова для всех реализаций родительского модуля.
2. В процессе реализации монтируемая схема задаётся разработчиком сервера и остаётся такой же стабильной, как информация библиотеки YANG для сервера.
3. Во время работы монтируемая схема задаётся данными экземпляра, которые являются частью монтируемой модели данных. При наличии нескольких экземпляров одной точки монтирования (например, в нескольких записях списка) монтируемые для каждого экземпляра модели данных могут различаться.

Определённый в этом документе механизм монтирования схемы поддерживает лишь два последних случая. Монтирование при разработке выходит за рамки документа и может быть рассмотрено в будущей версии языка YANG.

Монтирование схемы применяется к модели данных и, в частности, не принимает каких-либо допущений об источнике данных экземпляра для монтируемых схем. Это можно реализовать с применением тех же инструментов, что и для остальной системы, или реализовать через запрос к какой-либо иной системе. В будущих спецификациях могут быть заданы механизмы для контроля и мониторинга реализации конкретных точек монтирования.

Время и способ создания конкртных экземпляров точек монтирования на сервере выходят за рамки документа и могут быть заданы в будущих спецификациях.

Этот документ разрешает монтировать лишь полные модели данных. Другие спецификации могут расширить этот подход, определяя дополнительные механизмы, такие как монтирование субиерархии модуля.

Модули YANG в этом документе соответствуют архитектуре хранилищ данных управления сетью (Network Management Datastore Architecture или NMDA) [RFC8342].

2. Термины и обозначения

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

Ниже указаны термины, определённые в [RFC7950] и не переопределяемые здесь:

- action;
- container;
- data node;
- list;
- RPC operation;
- schema node;
- schema tree.

Ниже указаны термины, определённые в [RFC8342] и не переопределяемые здесь:

- client;
- notification;
- operational state;
- server

Термин system-controlled interface определён в [RFC8343] и не переопределяется здесь.

Термин YANG library content identifier определён в [RFC8525] и не переопределяется здесь.

Ниже приведены дополнительные определения применяемых в документе терминов.

mount point - точка монтирования
: Контейнер (container) или список (list), определения которого содержат оператор расширения mount-point. Аргумент оператора mount-point задаёт метку для точки монтирования.

schema - схема
: Набор деревьев схемы с общим корнем.

top-level schema - схема верхнего уровня
: Схема с корнем в корневом узле.

mounted schema - смонтированная схема
: Схема с корнем в точке монтирования.

parent schema (of a mounted schema) - родительская схема (смонтированной схемы)
: Схема, содержащая точку монтирования.

schema mount - монтирование схемы
: Механизм объединения моделей данных, определённый в этом документе.

2.1. Диаграммы деревьев

В диаграммах деревьев в этом документе применяются обозначения, заданные в [RFC8340].

2.2. Префиксы пространств имен

В этом документе имена узлов данных, расширения YANG, действия и другие элементы модели данных зачастую указываются без префикса, когда модуль YANG, где они определены, ясен из контекста. В остальных случаях применяется стандартный префикс, связанный с соответствующим модулем YANG, как показано в таблице 1.

Таблица 1. Префиксы пространств имен.

Префикс	***Модуль YANG***	***Документ***
yangmnt	ietf-yang-schema-mount	Раздел 9
inet	ietf-inet-types	[RFC6991]
yang	ietf-yang-types	[RFC6991]
yanglib	ietf-yang-library	[RFC7895], [RFC8525]

3. Монтирование схемы

Определённое здесь монтирование схемы предоставляет новый механизм расширения для использования с YANG 1.1 [RFC7950]. В этличие от имеющихся механизмов, упомянутых в разделе 1, монтирование схемы задаёт взаимодействие между исходным и целевым модулем YANG за пределами этих модулей.

3.1. Определение точки монтирования

Узел контейнера или списка становится точкой монтирования, если в его определении применён оператор расширения mount-point (задан в модуле ietf-yang-schema-mount). Это расширение может присутствовать лишь внутри операторов container и list.

Аргументом оператора mount-point является идентификатор YANG, указывающий метку точки монтирования. Модуль может включать несколько операторов mount-point с одинаковым аргументом.

Решение о размещении точки монтирования принимает разработчик родительской схемы. Точка монтирования может быть условной, если контейнер или список с точкой монтирования содержат оператор if-feature и/или when, представляющий эту точку.

Оператор расширения mount-point **недопустимо** применять в модулях YANG версии 1 [RFC6020], поскольку в таких случаях невозможно будет вызвать примонтированные операции RPC и получить примонтированные уведомления (см. 5. Операции RPC и уведомления). Отднако примонтированы могут быть модули с любой версией YANG (включая 1).

Отметим, что оператор расширения mount-point не создаёт нового узла данных.

3.2. Модель данных

Этот документ определяет модуль YANG 1.1 [RFC7950] ietf-yang-schema-mount с показанной ниже структурой.

```
module: ietf-yang-schema-mount
  +--ro schema-mounts
     +--ro namespace* [prefix]
     |  +--ro prefix    yang:yang-identifier
     |  +--ro uri?      inet:uri
     +--ro mount-point* [module label]
        +--ro module                yang:yang-identifier
        +--ro label                 yang:yang-identifier
        +--ro config?               boolean
        +--ro (schema-ref)
           +--:(inline)
           |  +--ro inline!
           +--:(shared-schema)
              +--ro shared-schema!
                 +--ro parent-reference*   yang:xpath1.0
```

3.3. Спецификация смонтированной схемы

Смонтированные схемы для всех точек монтирования в родительской схеме определяются из данных состояния в контейнере /schema-mounts.

Как правило модули, монтируемые в указанной точке, не связаны с модулями в родительской схеме. В частности, монтируемый модуль может (но не обязан) присутствовать в родительской схеме. Если модуль имеется в родительской схеме, он обычно не связан с данными родителя. Исключения возможны и должны определяться в самой модели, например, в [RFC8530] определён механизм привязки интерфейсов к смонтированным логическим элементам.

Контейнер /schema-mounts включает список mount-point в качестве одного из потомков. Каждая запись списка указывает (по ключу) точку монтирования и задаёт монтируемую схему.

Если точка монтирования задана в родительской схеме, но не имеет записи в списке mount-point, монтируемая схема отсутствует, т. е. экземплярам этой точки монтирования НЕДОПУСТИМО включать какие-либо данные, кроме определенных в родительской схеме.

Если в одном модуле задано несколько одноимённых точек монтирования (наприямую или путём определения в группировке, применяемой неоднократно), соответствующая запись mount-point применяется ко всем таким точкам.

Свойство config в узлах монтируемой схемы переопределяется и все узлы монтируемой схемы делаются доступными лишь для чтения (config false), если выполняется хотя бы одно из условий для точки монтирования:

- точка монтирования задана с config false;
- для листа config в соответствующей записи списка mount-point установлено значение false.

Запись списка mount-point может указывать монтируемую схему двумя способами inline (встроенная) или shared-schema (схема общего пользования).

Монтируемая схема определяется во время исполнения, каждый экземпляр точки монтирования, имеющийся в рабочем состоянии, **должен** содержать копию библиотечных данных YANG, определяющих монтируемую схему так же, как и схемы верхнего уровня. Предполагается, что клиент извлечёт эти данные из дерева экземпляров. В случае inline экземпляры одной точки монтирования **могут** применять разные схемы монтирования, тогда как для случая shared-schema все экземпляры **должны** использовать одну схему. Это значит, что в режиме shared-schema все экземпляры точки монтирования **должны** иметь один идентификатор содержимого библиотеки YANG. В случае inline применение двумя экземпляра одного идентификатора содержимого библиотеки YANG не гарантируется совпадение содержимого для этих экземпляров библиотеки YANG.

Примеры inline и shared-schema представлены в приложениях A.2 и A.3, соответственно.

3.4. Несколько уровней монтирования схем

Модули YANG в смонтированной схеме могут включать свои точки монтирования, в которых могут монтироваться другие схемы. Таким образом, можно создавать модели с произвольным числом примонтированных схем. Схему для точки монтирования в примонтированном модуле можно задать путём реализации модулей ietf-yang-library и ietf-yang-schema-mount в смонтированной схеме и заданием схем также, как описано выше для монтирования верхнего уровня.

4. Ссылка на узлы данных в родительской схеме

Фундаментальным принципом монтирования схем является работа смонтированной схемы точно так же, как схемы верхнего уровня, как будто та помещена в mount jail (тюрьма монтирования). Это означает, что все пути в примонтированной схеме (в leafref, instance-identifier, выражениях XPath [XPATH] и целевых узлах операторов augment) интерпретируются с корневым узлом в точке монтирования. Модули YANG смонтированной схемы, а также соответствующие данные экземпляра в результате не могут ссылаться на узлы схемы или данные экземпляров за переделами mount jail.

Однако такое ограничение иной раз слишком жёстко. Типичным примером служат экземпляры сетей (NI) [RFC8529], где каждый NI имеет свою машину маршрутизации, но список интерфейсов является глобальным и используется всеми NI. Если смоделировать это с монтированием схемы NI, общая схема будет иметь вид

```
+--rw interfaces
|  +--rw interface* [name]
|     ...
+--rw network-instances
   +--rw network-instance* [name]
      +--rw name
      +--mp root
         +--rw routing
            ...
```

Здесь контейнер root является точкой монтирования для схемы NI. Конфигурация маршрутизации внутри NI зачастую должна ссылаться на интерфейсы (по меньшей мере назначенные NI), что невозможно, если такая ссылка не может указывать узел в родительской схеме (имя интерфейса). Поэтому монтирование схемы допускает такие ссылки. Для каждой точки монтирования в случае shared-schema можно задать leaf-list с именем parent-reference, который может содержать выражения XPath 1.0. Каждое из таких выражений оценивается с узлом в родительском дереве данных, где точка монтирования задана как узел контекста. Результатом оценки **должен** быть набор node-set (см. описание узла parent-reference, где полностью определён контекст оценки). Для целей оценки выражений XPath внутри смонтированного дерева данных объединение всех таких node-set добавляется к доступному дереву данных.

Следует подчеркнуть, что узлы, указанные в parent-reference (leaf-list) доступны в смонтированной схеме лишь для оценки XPath. В частности, доступ к ним из смонтированной схемы невозможен по протоколам управления сетью, таким как NETCONF [RFC6241] и RESTCONF [RFC8040].

5. Операции RPC и уведомления

Если смонтированный модуль YANG задаёт операцию RPC, клиенты могут вызывать эту операцию, как будто она определена как действие для соответствующей точки монтирования (см. параграф 7.15 в [RFC7950]). Пример этого представлен в приложении A.4. Вызов операций RPC

Если сервер отправляет уведомление, заданное на верхнем уровне какого-либо примонтированного модуля, оно **должно** быть представленным как соединенное с точкой монтирования (см. параграф 7.16 в [RFC7950]).

Отметим, что встроенные (inline) действия и уведомления не будут работать, если они содержатся внутри списка (list) без оператора key (см. параграфы 7.15 и 7.16 в [RFC7950]). Поэтому, чтобы быть полезными, соделям с RPC, action и notification **не следует** иметь какого-либо предка, являющегося списком без оператора key. Это требование относится к определениям модулей, использующих оператор расширения mount-point.

6. Взаимодействие с NMDA

Решение для монтирования схем, представленное в этом документе, рассчитано на работу с серверами, реализующими NMDA [RFC8342] и старыми серверами, не поддерживающими NMDA. В частности, не поддерживающий NMDA сервер **может** реализовать выпуск 2016-06-21 модуля ietf-yang-library [RFC7895] в точке монтирования. Сервер с поддержкой NMDA MUST **должен** реализовать выпуск 2019-01-04 (или более свежий) модуля ietf-yang-library [RFC8525] в точке монтирования.

7. Взаимодействие с NACM

Если на сервере реализована модель управления доступом к конфигурации сети (Network Configuration Access Control Model или NACM) [RFC8341], она служит для контроля доступа к узлам, определенным примонтированной схемой так же, как для узлов схемы верхнего уровня. Предположим, например, что модуль ietf-interfaces смонтирован в контейнере root списка logical-network-element, определённого в [RFC8530]. Тогда можно использовать приведённый ниже путь NACM для управления доступом к контейнеру interfaces (символ \ служит для разделения длинной строки на части).

```
<path xmlns:lne=
        "urn:ietf:params:xml:ns:yang:ietf-logical-network-element"
      xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces">
  /lne:logical-network-elements\
    /lne:logical-network-element/lne:root/if:interfaces
</path>
```

8. Замечания для разработчиков

Сетевое управление устройствами, использующими модель данных с монтированием схемы, можно реализовать разными способами. В качестве типовых предусмотрены указанные ниже варианты.

Совместное управление

Данные экземпляра родительской и смонтированной схемы доступные в одном сеансе управления.

Раздельное управление

Одна (ведущая) сессия управления имеет доступ к данным экземпляра родительской и смонтированной схемы, а дополнительная сессия для каждого экземпляра точки монтирования имеет доступ лишь к своему примонтированному дереву.

9. Модуль YANG Schema Mount

Этот модуль ссылается на [RFC6991] и [RFC7950].

```
<CODE BEGINS> file "ietf-yang-schema-mount@2019-01-14.yang"

module ietf-yang-schema-mount {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-yang-schema-mount";
  prefix yangmnt;
```

```
import ietf-inet-types {
  prefix inet;
  reference
    "RFC 6991: Common YANG Data Types";
}

import ietf-yang-types {
  prefix yang;
  reference
    "RFC 6991: Common YANG Data Types";
}

organization
  "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

contact
  "WG Web:   <https://datatracker.ietf.org/wg/netmod/>
   WG List:  <mailto:netmod@ietf.org>

   Editor:   Martin Bjorklund
             <mailto:mbj@tail-f.com>

   Editor:   Ladislav Lhotka
             <mailto:lhotka@nic.cz>";

description
  "Этот модуль определяет оператор расширения YANG, который может
   служить для встраивания в модуль моделей данных, определённых в
   другом модуле YANG. Он также определяет данные рабочего
   состояния, которые задают общую структуру модели данных.

   Ключевые слова ДОЛЖНО, НЕДОПУСТИМО, ТРЕБУЕТСЯ, НУЖНО, НЕ СЛЕДУЕТ,
   СЛЕДУЕТ, НЕ НУЖНО, РЕКОМЕДУЕТСЯ, НЕ РЕКОМЕНДУЕТСЯ, МОЖНО и
   НЕОБЯЗАТЕЛЬНО в этом документе интерпретируются в соответствии с
   BCP 14 (RFC 2119) (RFC 8174) тогда и только тогда, когда они
   указаны заглавными буквами, как показано здесь.

   Авторские права (Copyright (c) 2019) принадлежат IETF Trust
   и лицам, указанным в качестве авторов кода. Все права защищены.

   Распространение и использование в исходной или двоичной форме с
   изменениями или без таковых разрешено в соответствии с лицензией
   Simplified BSD, изложенной в разделе 4  IETF Trust's Legal
   Provisions применительно к документам IETF
   (http://trustee.ietf.org/license-info).

   Эта версия данного модуля YANG является частью RFC 8528, где
   правовые вопросы рассмотрены более полно.";

revision 2019-01-14 {
  description
    "Исходный выпуск.";
  reference
    "RFC 8528: YANG Schema Mount";
}

/*
 * Расширения
 */

extension mount-point {
  argument label;
  description
    "Аргумент label является идентификатором YANG, т. е. имеет тип
     yang:yang-identifier.

     Оператор mount-point НЕДОПУСТИМО применять в модулях YANG
     версии 1, ни явно, ни через оператор uses.

     Оператор mount-point МОЖЕТ присутствовать как субоператор
     в container и list и НЕДОПУСТИМО его включение в другие места.
     НЕДОПУСТИМО включение более одного оператора mount-point в
     данный оператор container или list.

     Если точка монтирования определена в группировке, её метка
     привязывается к модулю, где группировка применяется.

     Точка монтирования задаёт место в иерархии узла, куда
     присоединяются другие модели данных. Сервер, реализующий
     модуль с точкой монтирования заполняет список
     /schema-mounts/mount-point подробными сведениями о моделях
     данных, монтируемых в каждой точке.

     Оператор mount-point не определяет нового узла данных.";
}
```

```
/*
 * Узлы данных состояния
 */

container schema-mounts {
  config false;
  description
    "Содержит сведения о структуре смонтированной модели данных,
     реализованной на сервере.";
  list namespace {
    key "prefix";
    description
      "Список сопоставления префиксов пространств имён, применяемых
       в выражениях XPath листьев parent-reference, с URI
       соответствующих пространств имен.";
    leaf prefix {
      type yang:yang-identifier;
      description
        "Префикс пространства имен.";
    }
    leaf uri {
      type inet:uri;
      description
        "URI пространства имен.";
    }
  }
  list mount-point {
    key "module label";
    description
      "Каждая запись списка задаёт схему для конкретной точки
       монтирования.

       Каждая точка монтирования ДОЛЖНА быть определена с
       использованием расширения mount-point в одном из модулей,
       указанных в серверном экземпляре библиотеки YANG с типом
       соответствия implement.";
    leaf module {
      type yang:yang-identifier;
      description
        "Имя модуля, содержащего точку монтирования.";
    }
    leaf label {
      type yang:yang-identifier;
      description
        "метка точки монтирования, определённой оператором
         mount-point.";
    }
    leaf config {
      type boolean;
      default "true";
      description
        "Если этот лист имеет значение false, все узлы данных
         смонтированной схемы будут доступны лишь для чтения
         (config false), независимо от их свойства config.";
    }
    choice schema-ref {
      mandatory true;
      description
        "Варианты для задания схемы.";
      container inline {
        presence
          "Полная, самодостаточная схема монтируется в точке.";
        description
          "Этот узел указывает, что сервер смонтировал по меньшей
           мере модуль ietf-yang-library в точке монтирования и его
           создание обеспечивает сведения о смонтированной схемы.

           В разных экземплярах точки монтирования могут
           монтироваться разные схемы.";
      }
      container shared-schema {
        presence
          "Смонтированная схема вместе с parent-reference
           составляют схему для этой точки монтирования.";
        description
          "Этот узел указывает, что сервер смонтировал по меньшей
           мере модуль ietf-yang-library в точке монтирования и его
           создание обеспечивает сведения о смонтированной схемы.
           При оценке выражения XPath в смонтированной схеме
           принимается во внимание лист-список parent-reference.

           Разные экземпляры точки монтирования ДОЛЖНЫ иметь одну
           и ту же смонтированную схему.";
        leaf-list parent-reference {
          type yang:xpath1.0;
```

```
                description
                  "Записями этого leaf-list служат выражения XPath 1.0,
                   которые оцениваются в указанном ниже контексте.

                   - Узлом контекста служит узел родительского дерева
                     данных, где определена точка mount-point.

                   - Доступным деревом является родительское дерево
                     данных без узлов, определённых в модулях,
                     смонтированных внутри родительской схемы.

                   - Положение и размер контекста равны 1.

                   - Набор привязок переменных пуст.

                   - Библиотекой функций служит библиотека функций ядра,
                     определённая в документе W3C XPath 1.0
                     (http://www.w3.org/TR/1999/REC-xpath-19991116) и
                     функции, определённые в разделе 10 RFC 7950.

                   - набор деклараций пространств имён определяется
                     списком namespace в иерархии schema-mounts.

                   Каждое выражение XPath ДОЛЖНО оцениваться в node-set
                   (возможно пустой). Для вычисления выражений XPath,
                   для которых узлы контекста заданы в смонтированной
                   схеме, объединение всех node-set и узлов родителя
                   добавляется в дерево доступных данных.

                   Отметим, что при монтировании самого модуля
                   ietf-yang-schema-mount узел parent-reference в
                   смонтированном модуле может ссылаться на узлы,
                   перенесенные в доступное дерево через parent-reference
                   в родительской схеме.";
              }
            }
          }
        }
      }
    }
    <CODE ENDS>
```

10. Взаимодействие с IANA

Этот документ регистрирует URI в реестре IETF XML Registry [RFC3688].

```
        URI: urn:ietf:params:xml:ns:yang:ietf-yang-schema-mount
        Registrant Contact: The IESG.
        XML: N/A, the requested URI is an XML namespace.
```

Документ также регистрирует модуль YANG в реестре YANG Module Names [RFC6020].

```
      name:         ietf-yang-schema-mount
      namespace:    urn:ietf:params:xml:ns:yang:ietf-yang-schema-mount
      prefix:       yangmnt
      reference:    RFC 8528
```

11. Вопросы безопасности

Заданный в этом документе модуль YANG определяет схему для данных, которая разработана для доступа через протоколы управления сетью, такие как NETCONF [RFC6241] и RESTCONF [RFC8040]. Нижним уровнем NETCONF является защищённый транспорт с обязательной реализацией Secure Shell (SSH) [RFC6242]. Нижним уровнем RESTCONF служит HTTPS с обязательной реализацией защищённого транспорта TLS [RFC8446].

Модель управления доступом NACM [RFC8341] обеспечивает средства, позволяющие предоставить доступ лишь конкретным пользователям NETCONF и RESTCONF к предопределённому подмножеству доступных в NETCONF или RESTCONF протокольных операций и содержимого.

Некоторые из доступных для чтения узлов этого модуля YANG могут быть конфиденциальными или уязвимыми в некоторых сетевых средах. Важно контролировать доступ к считыванию таких узлов данных (например, через операции get, get-config, notification). Ниже указаны ветви и узлы данных конфиденциальные или уязвимые в плане их чтения.

/schema-mounts

Схема, задаваемая этими данными состояния предоставляет подробные сведения о реализации сервера, которые могут помочь злоумышленнику определить возможности сервера и реализации с известными ошибками. Уязвимости сервера могут быть связаны с конкретными модулями, включёнными в схему, выпусками и функциями (feature) и даже отклонениями модулей. Например, если определённая операция над конкретным узлом данных заведомо вызывает аварию на сервере или существенному снижению производительности устройства, сведения схемы могут помочь атакующему идентифицировать сервер с таким дефектом, чтобы организовать DoS[1]-атаку на устройство.

Важно учитывать соображения безопасности для всех узлов монтируемых схем и контролировать доступ к этим узлам с использованием механизмов, описанных в разделе 7. Взаимодействие с NACM.

Необходимо соблюдать осторожность при создании выражений XPath parent-reference, поскольку результат оценки этих выражений добавляется в доступное дерево для любого выражения XPath найденного в смонтированной схеме.

[1]Denial-of-service - отказ в обслуживании.

12. Литература

12.1. Нормативные документы

[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>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[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>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

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

[RFC7895] Bierman, A., Bjorklund, M., and K. Watsen, "YANG Module Library", RFC 7895, DOI 10.17487/RFC7895, June 2016, <https://www.rfc-editor.org/info/rfc7895>.

[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>.

[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>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8525] Bierman, A., Bjorklund, M., Schoenwaelder, J., Watsen, K., and R. Wilton, "YANG Library", RFC 8525, DOI 10.17487/RFC8525, March 2019, <https://www.rfc-editor.org/info/rfc8525>.

[XPATH] Clark, J. and S. DeRose, "XML Path Language (XPath) Version 1.0", World Wide Web Consortium Recommendation REC-xpath-19991116, November 1999, <http://www.w3.org/TR/1999/REC-xpath-19991116>.

12.2. Дополнительная литература

[DEVICE-YANG] Lindem, A., Ed., Berger, L., Ed., Bogdanovic, D., and C. Hopps, "Network Device YANG Logical Organization", Work in Progress, draft-ietf-rtgwg-device-model-02, March 2017.

[IS-IS-YANG] Litkowski, S., Yeung, D., Lindem, A., Zhang, J., and L. Lhotka, "YANG Data Model for IS-IS protocol", Work in Progress[1], draft-ietf-isis-yang-isis-cfg-34, January 2019.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[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>.

[RFC8529] Berger, L., Hopps, C., Lindem, A., Bogdanovic, D., and X. Liu, "YANG Data Model for Network Instances", RFC 8529, DOI 10.17487/RFC8529, March 2019, <https://www.rfc-editor.org/info/rfc8529>.

[RFC8530] Berger, L., Hopps, C., Lindem, A., Bogdanovic, D., and X. Liu, "YANG Model for Logical Network Elements", RFC 8530, DOI 10.17487/RFC8530, March 2019, <https://www.rfc-editor.org/info/rfc8530>.

[YANG-MOUNT] Clemm, A., Voit, E., and J. Medved, "Mounting YANG-Defined Information from Remote Datastores", Work in Progress, draft-clemm-netmod-mount-06, March 2017.

Приложение A. Пример модели устройства с LNE и NI

Этот ненормативный пример показывает реализацию модели устройства (раздел 2 [DEVICE-YANG]), использующего логические элементы сети (logical network element или LNE) и экземпляры сетей (network instance или NI).

В примерах ниже символ \ служит для разделения на части длинных строк.

A.1. Физическое устройство

Модель данных физического устройства можно описать содержимым библиотеки YANG в предположении поддержки сервером архитектуры NMDA.

```
{
   "ietf-yang-library:yang-library": {
```

[1] Опубликовано в RFC 9130. *Прим. перев.*

```
      "content-id": "14e2ab5dc325f6d86f743e8d3ade233f1a61a899",
      "module-set": [
        {
          "name": "physical-device-modules",
          "module": [
            {
              "name": "ietf-datastores",
              "revision": "2018-02-14",
              "namespace":
                "urn:ietf:params:xml:ns:yang:ietf-datastores"
            },
            {
              "name": "iana-if-type",
              "revision": "2015-06-12",
              "namespace": "urn:ietf:params:xml:ns:yang:iana-if-type"
            },
            {
              "name": "ietf-interfaces",
              "revision": "2018-02-20",
              "feature": ["arbitrary-names", "pre-provisioning" ],
              "namespace":
                "urn:ietf:params:xml:ns:yang:ietf-interfaces"
            },
            {
              "name": "ietf-ip",
              "revision": "2018-02-22",
              "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip"
            },
            {
              "name": "ietf-logical-network-element",
              "revision": "2018-03-20",
              "feature": [ "bind-lne-name" ],
              "namespace":
                "urn:ietf:params:xml:ns:yang:\
                ietf-logical-network-element"
            },
            {
              "name": "ietf-yang-library",
              "revision": "2019-01-04",
              "namespace":
                "urn:ietf:params:xml:ns:yang:ietf-yang-library"
            },
            {
              "name": "ietf-yang-schema-mount",
              "revision": "2019-01-14",
              "namespace":
                "urn:ietf:params:xml:ns:yang:ietf-yang-schema-mount"
            }
          ],
          "import-only-module": [
            {
              "name": "ietf-inet-types",
              "revision": "2013-07-15",
              "namespace":
                "urn:ietf:params:xml:ns:yang:ietf-inet-types"
            },
            {
              "name": "ietf-yang-types",
              "revision": "2013-07-15",
              "namespace":
                "urn:ietf:params:xml:ns:yang:ietf-yang-types"
            }
          ]
        }
      ],
      "schema": [
        {
          "name": "physical-device-schema",
          "module-set": [ "physical-device-modules" ]
        }
      ],
      "datastore": [
        {
          "name": "ietf-datastores:running",
          "schema": "physical-device-schema"
        },
        {
          "name": "ietf-datastores:operational",
          "schema": "physical-device-schema"
        }
      ]
    }
  }
```

A.2. Логические элементы сети

Каждый элемент LNE может иметь конкретную модель данных, которая определяется во время работы, поэтому приемлемо монтирование по методу inline. В результате применяются указанные ниже данные schema-mounts.

```
{
  "ietf-yang-schema-mount:schema-mounts": {
    "mount-point": [
      {
        "module": "ietf-logical-network-element",
        "label": "root",
        "inline": {}
      }
    ]
  }
}
```

Администратор хоста с устройством настраивает запись для каждого экземпляра LNE, например,

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "enabled": true,
        "ietf-logical-network-element:bind-lne-name": "eth0"
      }
    ]
  },
  "ietf-logical-network-element:logical-network-elements": {
    "logical-network-element": [
      {
        "name": "lne-1",
        "managed": true,
        "description": "LNE with NIs",
        "root": {
          ...
        }
      }
      ...
    ]
  }
}
```

а затем помещает требуемые данные состояния как содержимое экземпляра root, куда следует включить хотя бы:

- данные библиотеки YANG, задающие модель данных LNE, например, предположение, что сервер не реализует NMDA

```
{
  "ietf-yang-library:modules-state": {
    "module-set-id": "9358e11874068c8be06562089e94a89e0a392019",
    "module": [
      {
        "name": "iana-if-type",
        "revision": "2014-05-08",
        "namespace": "urn:ietf:params:xml:ns:yang:iana-if-type",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-inet-types",
        "revision": "2013-07-15",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types",
        "conformance-type": "import"
      },
      {
        "name": "ietf-interfaces",
        "revision": "2014-05-08",
        "feature": [
          "arbitrary-names",
          "pre-provisioning"
        ],
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-interfaces",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-ip",
        "revision": "2014-06-16",
        "feature": [
          "ipv6-privacy-autoconf"
        ],
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-network-instance",
        "revision": "2018-03-20",
        "feature": [
```

```
            "bind-network-instance-name"
          ],
          "namespace":
            "urn:ietf:params:xml:ns:yang:ietf-network-instance",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-library",
          "revision": "2016-06-21",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-library",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-schema-mount",
          "revision": "2019-01-14",
          "namespace":
            "urn:ietf:params:xml:ns:yang:ietf-yang-schema-mount",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-types",
          "revision": "2013-07-15",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types",
          "conformance-type": "import"
        }
      ]
    }
  }
```

- данные состояния для интерфейсов, назначенных экземпляру LNE (которые фактически становятся управляемыми системой интерфейсами для LNE), например,

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "oper-status": "up",
        "statistics": {
          "discontinuity-time": "2016-12-16T17:11:27+02:00"
        },
        "ietf-ip:ipv6": {
          "address": [
            {
              "ip": "fe80::42a8:f0ff:fea8:24fe",
              "origin": "link-layer",
              "prefix-length": 64
            }
          ]
        }
      }
    ]
  }
}
```

A.3. Экземпляры сетей

В предположении, что экземпляры сетей используют общую модель данных, их можно монтировать по методу shared-schema, как показано ниже.

```
{
  "ietf-yang-schema-mount:schema-mounts": {
    "namespace": [
      {
          "prefix": "if",
          "uri": "urn:ietf:params:xml:ns:yang:ietf-interfaces"
      },
      {
          "prefix": "ni",
          "uri": "urn:ietf:params:xml:ns:yang:ietf-network-instance"
      }
    ],
    "mount-point": [
      {
        "module": "ietf-network-instance",
        "label": "root",
          "shared-schema": {
            "parent-reference": [
              "/if:interfaces/if:interface[\
              ni:bind-network-instance-name = current()/../ni:name]"
            ]
          }
      }
    ]
  }
}
```

Отметим, что модуль ietf-interfaces появляется в листе-списке parent-reference для смонтированной схемы NI. Это значит, что ссылки на интерфейсы LNE, такие как outgoing-interface в статических маршрутах, будут действительны, несмотря на то, что ietf-interfaces не является частью схемы NI.

A.4. Вызов операций RPC

Предположим, что смонтированная модель данных NI реализует модуль ietf-isis [IS-IS-YANG]. Операция RPC, заданная в этом модуле (например, clear-adjacency), может быть вызвана в клиентской сессии с сервером RESTCONF этого LNE, как действие, привязанное к точке монтирования конкретного экземпляра сети с использованием URI запроса (в одну строку) вида

```
POST /restconf/data/ietf-network-instance:network-instances/
    network-instance=rtrA/root/ietf-isis:clear-adjacency HTTP/1.1
```

Участники работы

Идея о каком-либо способе объединения схем из разных модулей YANG была независимо предложена рядом людей.

- Авторы [YANG-MOUNT]:
  - Lou Berger, LabN Consulting, L.L.C., <lberger@labn.net>;
  - Alexander Clemm, Huawei, <alexander.clemm@huawei.com>;
  - Christian Hopps, Deutsche Telekom, <chopps@chopps.org>.
- Jan Medved, Cisco, <jmedved@cisco.com>.
- Eric Voit, Cisco, <evoit@cisco.com>.

Адреса авторов

Martin Bjorklund
Tail-f Systems
Email: mbj@tail-f.com

Ladislav Lhotka
CZ.NIC
Email: lhotka@nic.cz

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru