

Internet Engineering Task Force (IETF)
Request for Comments: 8344
Obsoletes: 7277
Category: Standards Track
ISSN: 2070-1721

М. Bjorklund
Tail-f Systems
March 2018

Модель данных YANG для управления IP

A YANG Data Model for IP Management

Аннотация

Этот документ определяет модель данных YANG для управления реализациями IP. Модель включает данные конфигурации и состояния системы.

Модель данных YANG в этом документе соответствует архитектуре хранилища данных сетевого управления NMDA¹, определенной в RFC 8342.

Документ отменяет действие RFC 7277.

Статус документа

Документ относится к категории Internet Standards Track.

Документ является результатом работы IETF² и представляет согласованный взгляд сообщества IETF. Документ прошёл открытое обсуждение и был одобрен для публикации IESG³. Не все одобренные IESG документы претендуют на статус Internet Standard (см. раздел 2 в RFC 7841).

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке <https://www.rfc-editor.org/info/rfc8344>.

Авторские права

Авторские права (Copyright (c) 2018) принадлежат IETF Trust и лицам, указанным в качестве авторов документа. Все права защищены.

К документу применимы права и ограничения, указанные в BCP 78 и IETF Trust Legal Provisions и относящиеся к документам IETF (<http://trustee.ietf.org/license-info>), на момент публикации данного документа. Прочтите упомянутые документы внимательно. Фрагменты программного кода, включённые в этот документ, распространяются в соответствии с упрощённой лицензией BSD, как указано в параграфе 4.e документа IETF Trust Legal Provisions, без каких-либо гарантий (как указано в Simplified BSD License).

Оглавление

1. Введение.....	1
1.1. Отличия от RFC 7277.....	2
1.2. Терминология.....	2
1.3. Диаграммы деревьев.....	2
2. Модель данных IP.....	2
3. Связь с IP-MIB.....	3
4. Модуль YANG для управления IP.....	3
5. Взаимодействие с IANA.....	13
6. Вопросы безопасности.....	13
7. Литература.....	14
7.1. Нормативные документы.....	14
7.2. Дополнительная литература.....	15
Приложение А. Пример отклика NETCONF <get-config>.....	15
Приложение В. Пример отклика NETCONF <get-data>.....	15
Благодарности.....	16
Адрес автора.....	16

1. Введение

Этот документ определяет модель данных YANG [RFC7950] для управления реализациями IP.

Модель данных охватывает конфигурацию параметров IPv4 и IPv6 на уровне интерфейсов, а также отображение адресов IP на адреса канального уровня. Модель также дает информацию об используемых в работе адресах IP и имеющихся отображениях канального уровня. Параметры на уровне интерфейсов добавлены путем дополнения (augmentation) модели данных интерфейса, определенной в [RFC8343].

Эта версия модели данных IP поддерживает архитектуру хранилищ NMDA [RFC8342].

¹Network Management Datastore Architecture.

²Internet Engineering Task Force - комиссия по решению инженерных задач Internet.

³Internet Engineering Steering Group - комиссия по инженерным разработкам Internet.

1.1. Отличия от RFC 7277

Ветви (subtree) ipv4 и ipv6 с узлами данных config false в ветви /interfaces-state/interface отменены. Все узлы данных config false сейчас присутствуют в ветвях ipv4 и ipv6 субдерева /interfaces/interface.

Серверы, не поддерживающие NMDA или желающие работать с клиентами, не поддерживающими NMDA, **могут** реализовать отмененные ветви ipv4 и ipv6 в субдереве /interfaces-state/interface.

1.2. Терминология

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.

Перечисленные ниже термины определены в [RFC8342] и не определяются здесь заново.

- client (клиент);
- server (сервер);
- configuration (конфигурация);
- system state (состояние системы);
- intended configuration (предполагаемая конфигурация);
- running configuration datastore (хранилище данных рабочей конфигурации);
- operational state (операционное состояние);
- operational state datastore (хранилище данных операционного состояния).

Перечисленные ниже термины определены в [RFC7950] и не определяются здесь заново.

- augment (дополнение, усиление);
- data model (модель данных);
- data node (узел данных).

Терминология для описания моделей YANG представлена в [RFC7950].

1.3. Диаграммы деревьев

Диаграммы деревьев, используемой в этом документе, следуют обозначениям, определенным в [RFC8340].

2. Модель данных IP

Этот документ определяет модуль YANG ietf-ip, который дополняет списки interface, определенные в модуле ietf-interfaces [RFC8343], связанными с IP узлами данных.

Структура узлов данных IP на уровне интерфейса, за исключением отмененных полей данных, приведена ниже.

```

module: ietf-ip
  augment /if:interfaces/if:interface:
    +---rw ipv4!
    | +---rw enabled?          boolean
    | +---rw forwarding?      boolean
    | +---rw mtu?             uint16
    | +---rw address* [ip]
    | | +---rw ip              inet:ipv4-address-no-zone
    | | +---rw (subnet)
    | | | +---:(prefix-length)
    | | | | +---rw prefix-length?  uint8
    | | | | +---:(netmask)
    | | | | +---rw netmask?        yang:dotted-quad
    | | | | {ipv4-non-contiguous-netmasks}?
    | | +---ro origin?          ip-address-origin
    | +---rw neighbor* [ip]
    | | +---rw ip              inet:ipv4-address-no-zone
    | | +---rw link-layer-address yang:phys-address
    | | +---ro origin?         neighbor-origin
    +---rw ipv6!
    +---rw enabled?            boolean
    +---rw forwarding?        boolean
    +---rw mtu?                uint32
    +---rw address* [ip]
    | +---rw ip                inet:ipv6-address-no-zone
    | +---rw prefix-length     uint8
    | +---ro origin?          ip-address-origin
    | +---ro status?          enumeration
    +---rw neighbor* [ip]
    | +---rw ip                inet:ipv6-address-no-zone
    | +---rw link-layer-address yang:phys-address
    | +---ro origin?          neighbor-origin
    | +---ro is-router?       empty
    | +---ro state?           enumeration
  
```

```

+--rw dup-addr-detect-transmits? uint32
+--rw autoconf
  +--rw create-global-addresses? boolean
  +--rw create-temporary-addresses? boolean
  |   {ipv6-privacy-autoconf}?
  +--rw temporary-valid-lifetime? uint32
  |   {ipv6-privacy-autoconf}?
  +--rw temporary-preferred-lifetime? uint32
      {ipv6-privacy-autoconf}?

```

Модель определяет для интерфейса два контейнера - ipv4 и ipv6, представляющие семейства адресов IPv4 и IPv6. В каждом контейнере имеется лист enabled, который показывает разрешено ли семейство адресов на данном интерфейсе, и лист forwarding, показывающий разрешена ли пересылка пакетов IP данного семейства на этом интерфейсе. В каждом контейнере имеется также список адресов и список отображений адресов IP на адреса канального уровня.

3. Связь с IP-MIB

Если устройство реализует IP-MIB [RFC4293], каждая запись в списках ipv4/address и ipv6/address отображается в один из объектов ipAddressEntry, где ipAddressIndex указывает address в объекте для интерфейса.

IP-MIB определяет объекты для управления сообщениями IPv6 Router Advertisement. Соответствующие узла данных YANG определены в [RFC8022].

Записи ipv4/neighbor и ipv6/neighbor отображаются на ipNetToPhysicalTable.

Ниже приведена таблица соответствия узлов данных YANG объектам IP-MIB.

Узлы данных YANG для интерфейса и связанные объекты IP-MIB.

Узел данных YANG в /if:interfaces/if:interface	Объект IP-MIB
ipv4	ipv4InterfaceEnableStatus
ipv4/enabled	ipv4InterfaceEnableStatus
ipv4/address	ipAddressEntry
ipv4/address/ip	ipAddressAddrType ipAddressAddr
ipv4/neighbor	ipNetToPhysicalEntry
ipv4/neighbor/ip	ipNetToPhysicalNetAddressType ipNetToPhysicalNetAddress
ipv4/neighbor/link-layer-address	ipNetToPhysicalPhysAddress
ipv4/neighbor/origin	ipNetToPhysicalType
ipv6	ipv6InterfaceEnableStatus
ipv6/enabled	ipv6InterfaceEnableStatus
ipv6/forwarding	ipv6InterfaceForwarding
ipv6/address	ipAddressEntry
ipv6/address/ip	ipAddressAddrType ipAddressAddr
ipv4/address/origin	ipAddressOrigin
ipv6/address/status	ipAddressStatus
ipv6/neighbor	ipNetToPhysicalEntry
ipv6/neighbor/ip	ipNetToPhysicalNetAddressType ipNetToPhysicalNetAddress
ipv6/neighbor/link-layer-address	ipNetToPhysicalPhysAddress
ipv6/neighbor/origin	ipNetToPhysicalType
ipv6/neighbor/state	ipNetToPhysicalState

4. Модуль YANG для управления IP

Этот модуль импортирует определения типов (typedef) из [RFC6991] и [RFC8343], а они ссылаются на [RFC791], [RFC826], [RFC4861], [RFC4862], [RFC4941], [RFC7217] и [RFC8200].

```

<CODE BEGINS> file "ietf-ip@2018-02-22.yang"
module ietf-ip {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-ip";
  prefix ip;

  import ietf-interfaces {
    prefix if;

```

```
}
import ietf-inet-types {
  prefix inet;
}
import ietf-yang-types {
  prefix yang;
}

organization
  "IETF NETMOD (Network Modeling) 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>";

description
  "Этот модуль содержит набор определений YANG для управления
  реализациями IP.

  Авторские права (Copyright (c) 2018) принадлежат IETF Trust
  и лицам, указанным как авторы кода. Все права защищены.

  Распространение и использование в исходной и двоичной форме
  с изменениями или без них разрешается в соответствии с условиями,
  указанными в упрощённой лицензии BSD, изложенной в разделе 4.с
  Правового положения IETF Trust применительно к документам IETF
  (http://trustee.ietf.org/license-info).

  Эта версия модуля YANG является частью RFC 8344, где
  правовые аспекты выражены более полно."

revision 2018-02-22 {
  description
    "Обновление для поддержки NMDA.";
  reference
    "RFC 8344: A YANG Data Model for IP Management";
}

revision 2014-06-16 {
  description
    "Исходный выпуск.";
  reference
    "RFC 7277: A YANG Data Model for IP Management";
}

/*
 * функции
 */

feature ipv4-non-contiguous-netmasks {
  description
    "Указывает поддержку масок подсетей с разрывами.";
}

feature ipv6-privacy-autoconf {
  description
    "Указывает поддержку расширений приватности для автоматической
    настройки IPv6 без поддержки состояний.";
  reference
    "RFC 4941: Privacy Extensions for Stateless Address
    Autoconfiguration in IPv6";
}

/*
 * Определения типов
 */

typedef ip-address-origin {
  type enumeration {
    enum other {
      description
        "Ничего из перечисленного ниже.";
    }
    enum static {
      description
        "Указывает, что адрес настроен статически, например с
        помощью NETCONF или командного интерфейса.";
    }
    enum dhcp {
      description
        "Указывает, что адрес выделен системе сервером DHCP.";
    }
    enum link-layer {
```

```

    description
        "Указывает адрес, созданный автоматической настройкой IPv6
        без поддержки состояния, который включает адрес link-layer
        в идентификатор интерфейса.";
    }
    enum random {
        description
            "Указывает, что адрес выбран системой случайно, например,
            временный адрес IPv4 из сети 169.254/16, как описано в
            RFC 4941, или семантически «непрозрачный» адрес, как
            описано в RFC 7217.";
        reference
            "RFC 4941: Privacy Extensions for Stateless Address
            Autoconfiguration in IPv6
            RFC 7217: A Method for Generating Semantically Opaque
            Interface Identifiers with IPv6 Stateless
            Address Autoconfiguration (SLAAC)";
    }
    }
    description
        "Источник адреса.";
}

typedef neighbor-origin {
    type enumeration {
        enum other {
            description
                "Ничего из указанного ниже.";
        }
        enum static {
            description
                "Указывает статически заданное отображение, например, с
                помощью NETCONF или командного интерфейса.";
        }
        enum dynamic {
            description
                "Указывает, что отображение распознано динамически,
                например, по протоколу IPv4 ARP или IPv6 Neighbor
                Discovery.";
        }
    }
    description
        "Источник записи о соседе.";
}

/*
 * Узлы данных
 */

augment "/if:interfaces/if:interface" {
    description
        "IP-параметры интерфейсов. Если интерфейс не способен
        поддерживать IP, серверу недопустимо разрешать клиенту
        настройку этих параметров.";

    container ipv4 {
        presence
            "Разрешает IPv4, если лист enabled (по умолчанию true)
            не имеет значения false";
        description
            "Параметры для семейства адресов IPv4.";

        leaf enabled {
            type boolean;
            default true;
            description
                "Управляет поддержкой IPv4 на этом интерфейсе. Если
                IPv4 разрешён, интерфейс подключается к стеку IPv4 и
                может передавать и принимать пакеты IPv4.";
        }
        leaf forwarding {
            type boolean;
            default false;
            description
                "Управляет пересылкой пакетов IPv4 с дейтаграммами, бу,
                принятыми интерфейсом, но не адресованными ему.
                Маршрутизаторы IPv4 пересылают дейтаграммы, хосты не
                пересылают (за исключением source-route через хост).";
        }
    }
    leaf mtu {
        type uint16 {
            range "68..max";
        }
        units "octets";
    }
}

```

```
description
  "Размер (в октетах) максимального пакета IPv4, который
  интерфейс будет передавать и принимать.

  Сервер может ограничивать разрешённые значения для этого
  листа в зависимости от типа интерфейса.

  Если лист не настроен, применяемое значение MTU зависит
  от типа интерфейса.";
reference
  "RFC 791: Internet Protocol";
}
list address {
  key "ip";
  description
    "Список адресов IPv4 на интерфейсе.";
  leaf ip {
    type inet:ipv4-address-no-zone;
    description
      "Адрес IPv4 на интерфейсе.";
  }
  choice subnet {
    mandatory true;
    description
      "Подсеть можно указать размером префикса или (если сервер
      поддерживает маски с разрывом) листом netmask.";
    leaf prefix-length {
      type uint8 {
        range "0..32";
      }
      description
        "Размер префикса подсети.";
    }
    leaf netmask {
      if-feature ipv4-non-contiguous-netmasks;
      type yang:dotted-quad;
      description
        "Подсеть задаётся маской.";
    }
  }
  leaf origin {
    type ip-address-origin;
    config false;
    description
      "Источник адреса.";
  }
}
list neighbor {
  key "ip";
  description
    "Список сопоставлений адресов IPv4 с адресами канального
    уровня.

    Записи этого списка в предполагаемой конфигурации
    служат в качестве статических записей кэша ARP.
    В рабочем состоянии этот список представляет кэш ARP.";
  reference
    "RFC 826: An Ethernet Address Resolution Protocol";

  leaf ip {
    type inet:ipv4-address-no-zone;
    description
      "Адрес IPv4 соседнего узла.";
  }
  leaf link-layer-address {
    type yang:phys-address;
    mandatory true;
    description
      "Адрес соседа на канальном уровне.";
  }
  leaf origin {
    type neighbor-origin;
    config false;
    description
      "Источник этой записи о соседе.";
  }
}
}
container ipv6 {
  presence
    "Разрешает IPv6, если лист enabled (по умолчанию true)
    не имеет значения false";
  description
    "Параметры для семейства адресов IPv6.";
```

```

leaf enabled {
  type boolean;
  default true;
  description
    "Управляет поддержкой IPv6 на этом интерфейсе. Если
    IPv6 разрешён, интерфейс подключается к стеку IPv4 и
    может передавать и принимать пакеты IPv6.";
}
leaf forwarding {
  type boolean;
  default false;
  description
    "Управляет пересылкой пакетов IPv6 с дейтаграммами, by,
    принятыми интерфейсом, но не адресованными ему.
    Маршрутизаторы IPv6 пересылают дейтаграммы, хосты не
    пересылают (за исключением source-route через хост).";
  reference
    "RFC 4861: Neighbor Discovery for IP version 6 (IPv6)
    Section 6.2.1, IsRouter";
}
leaf mtu {
  type uint32 {
    range "1280..max";
  }
  units "octets";
  description
    "Размер (в октетах) максимального пакета IPv6, который
    интерфейс будет передавать и принимать.

    Сервер может ограничивать разрешённые значения для этого
    листа в зависимости от типа интерфейса.

    Если лист не настроен, применяемое значение MTU зависит
    от типа интерфейса.";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
    Specification
    Section 5";
}
list address {
  key "ip";
  description
    "Список адресов IPv6 на интерфейсе.";

  leaf ip {
    type inet:ipv6-address-no-zone;
    description
      "Адрес IPv6 на интерфейсе.";
  }
  leaf prefix-length {
    type uint8 {
      range "0..128";
    }
    mandatory true;
    description
      "Размер префикса подсети.";
  }
  leaf origin {
    type ip-address-origin;
    config false;
    description
      "Источник адреса.";
  }
  leaf status {
    type enumeration {
      enum preferred {
        description
          "Это действительный адрес, который может указывать
          отправителя или получателя пакета.";
      }
      enum deprecated {
        description
          "Это действительный, но устаревший адрес, который не
          следует применять для отправителя в новых пакетах,
          но следует корректно обрабатывать в получаемых.";
      }
      enum invalid {
        description
          "Это недействительный адрес и его не следует
          указывать в поле отправителя или получателя пакета";
      }
      enum inaccessible {
        description
          "Адрес недоступен, поскольку интерфейс, в котором он

```

```
        связан, не работает.";
    }
    enum unknown {
        description
            "Статус невозможно определить по какой-то причине.";
    }

    enum tentative {
        description
            "Уникальность адреса на канале проверяется. Адреса с
            таким статусом не следует применять для общих
            коммуникаций, оставляя его лишь для проверки
            уникальности адреса.";
    }
    enum duplicate {
        description
            "Адрес сочтён неуникальным на канале и его
            недопустимо использовать.";
    }
    enum optimistic {
        description
            "Адрес доступен для использования с учётом
            ограничений, уникальность на канале проверяется.";
    }
}
config false;
description
    "Статус адреса. Большинство состояний соответствует
    значениям из IPv6 Stateless Address Autoconfiguration.";
reference
    "RFC 4293: Management Information Base for the
    Internet Protocol (IP)
    - IpAddressStatusTC
    RFC 4862: IPv6 Stateless Address Autoconfiguration";
}
}

list neighbor {
    key "ip";
    description
        "Список сопоставлений адресов IPv6 с адресами канального
        уровня.
        Записи этого списка в предполагаемой конфигурации
        служат в качестве статических записей кэша соседей.
        В рабочем состоянии список представляет кэш соседей.";
    reference
        "RFC 4861: Neighbor Discovery for IP version 6 (IPv6)";

    leaf ip {
        type inet:ipv6-address-no-zone;
        description
            "Адрес IPv6 соседнего узла.";
    }
    leaf link-layer-address {
        type yang:phys-address;
        mandatory true;
        description
            "Адрес соседнего узла на канальном уровне.
            В рабочем состоянии при state incomplete этот лист
            не создается.";
    }
    leaf origin {
        type neighbor-origin;
        config false;
        description
            "Источник записи о соседе.";
    }
    leaf is-router {
        type empty;
        config false;
        description
            "Указывает, что сосед служит маршрутизатором.";
    }
}

leaf state {
    type enumeration {
        enum incomplete {
            description
                "Распознавание адреса не завершено и адрес канального
                уровня для соседа ещё не определён.";
        }
        enum reachable {
            description
                "Грубо говоря, сосед был доступен недавно
                (десятки секунд назад).";
        }
    }
}
```

```

enum stale {
  description
    "О доступности соседа больше нет сведений, но пока
    трафик передаётся ему, не следует проверять его
    доступность.";
}
enum delay {
  description
    "О доступности соседа больше нет сведений и трафик
    недавно передавался ему. Проверку его доступности
    тестовыми пакетами следует отложить на короткое
    время, чтобы протоколы верхних уровней могли
    предоставить подтверждение доступности.";
}
enum probe {
  description
    "О доступности соседа больше нет сведений и для её
    проверки передаются пакеты Neighbor Solicitation.";
}
}
config false;
description
  "Состояние Neighbor Unreachability Detection для записи";
reference
  "RFC 4861: Neighbor Discovery for IP version 6 (IPv6)
  Section 7.3.2";
}
}

leaf dup-addr-detect-transmits {
  type uint32;
  default 1;
  description
    "Число последовательных сообщений Neighbor Solicitation,
    переданных в процессе Duplicate Address Detection на
    предварительный адрес. Нулевое значение указывает, что
    Duplicate Address Detection не выполняется для
    предварительных адресов. Значение 1 указывает одиночную
    отправку без повторов.";
  reference
    "RFC 4862: IPv6 Stateless Address Autoconfiguration";
}

container autoconf {
  description
    "Параметры управления автоматической настройкой адресов
    IPv6 в соответствии с RFC 4862.";
  reference
    "RFC 4862: IPv6 Stateless Address Autoconfiguration";

  leaf create-global-addresses {
    type boolean;
    default true;
    description
      "При значении true хост создаёт глобальные адреса в
      соответствии с RFC 4862.";
    reference
      "RFC 4862: IPv6 Stateless Address Autoconfiguration
      Section 5.5";
  }

  leaf create-temporary-addresses {
    if-feature ipv6-privacy-autoconf;
    type boolean;
    default false;
    description
      "При значении true хост создаёт временные адреса в
      соответствии с RFC 4941.";
    reference
      "RFC 4941: Privacy Extensions for Stateless Address
      Autoconfiguration in IPv6";
  }

  leaf temporary-valid-lifetime {
    if-feature ipv6-privacy-autoconf;
    type uint32;
    units "seconds";
    default 604800;
    description
      "Срок действия временных адресов.";
    reference
      "RFC 4941: Privacy Extensions for Stateless Address
      Autoconfiguration in IPv6
      - TEMP_VALID_LIFETIME";
  }

  leaf temporary-preferred-lifetime {
    if-feature ipv6-privacy-autoconf;
    type uint32;
    units "seconds";
  }
}

```

```

    default 86400;
    description
        "Срок предпочтения временных адресов.";
    reference
        "RFC 4941: Privacy Extensions for Stateless Address
        Autoconfiguration in IPv6
        - TEMP_PREFERRED_LIFETIME";
    }
}
}
}
/*
 * Унаследованные узлы данных операционных состояний
 */

augment "/if:interfaces-state/if:interface" {
    status deprecated;
    description
        "Узлы данных рабочего состояния IP на интерфейсах.";

    container ipv4 {
        presence
            "Присутствует при включении IPv4 на интерфейсе.";
        config false;
        status deprecated;
        description
            "Зависящие от интерфейса параметры для семейства IPv4.";

        leaf forwarding {
            type boolean;
            status deprecated;
            description
                "Указывает, включена ли на интерфейсе пересылка IPv4.";
        }
        leaf mtu {
            type uint16 {
                range "68..max";
            }
            units "octets";
            status deprecated;
            description
                "Максимальный размер пакета IPv4, который интерфейс может
                передать и принять (в октетах).";
            reference
                "RFC 791: Internet Protocol";
        }
    }

    list address {
        key "ip";
        status deprecated;
        description
            "Список адресов IPv4 на интерфейсе.";

        leaf ip {
            type inet:ipv4-address-no-zone;
            status deprecated;
            description
                "Адрес IPv4 на интерфейсе.";
        }
    }

    choice subnet {
        status deprecated;
        description
            "Подсеть можно указать размером префикса или (если сервер
            поддерживает маски с разрывом) листом netmask.";
        leaf prefix-length {
            type uint8 {
                range "0..32";
            }
            status deprecated;
            description
                "Размер префикса подсети.";
        }
        leaf netmask {
            if-feature ipv4-non-contiguous-netmasks;
            type yang:dotted-quad;
            status deprecated;
            description
                "Подсеть задаётся как netmask.";
        }
    }
}
leaf origin {
    type ip-address-origin;
    status deprecated;
    description

```

```

    "Источник данного адреса.";
  }
}
list neighbor {
  key "ip";
  status deprecated;
  description
    "Список сопоставлений адресов IPv4 с адресами канального
    уровня. Этот список представляет ARP Cache.";
  reference
    "RFC 826: An Ethernet Address Resolution Protocol";

  leaf ip {
    type inet:ipv4-address-no-zone;
    status deprecated;
    description
      "Адрес IPv4 у соседнего узла.";
  }
  leaf link-layer-address {
    type yang:phys-address;
    status deprecated;
    description
      "Адрес канального уровня у соседнего узла.";
  }
  leaf origin {
    type neighbor-origin;
    status deprecated;
    description
      "Источник этой записи о соседе.";
  }
}
}

container ipv6 {
  presence
    "Присутствует при включении IPv6 на интерфейсе.";
  config false;
  status deprecated;
  description
    "Параметры для семейства адресов IPv6.";

  leaf forwarding {
    type boolean;
    default false;
    status deprecated;
    description
      "Указывает, включена ли на интерфейсе пересылка IPv6.";
    reference
      "RFC 4861: Neighbor Discovery for IP version 6 (IPv6)
      Section 6.2.1, IsRouter";
  }
  leaf mtu {
    type uint32 {
      range "1280..max";
    }
    units "octets";
    status deprecated;
    description
      "Максимальный размер пакета IPv6, который интерфейс может
      передать и принять (в октетах).";
    reference
      "RFC 8200: Internet Protocol, Version 6 (IPv6)
      Specification
      Section 5";
  }
}

list address {
  key "ip";
  status deprecated;
  description
    "Список адресов IPv6 на интерфейсе.";

  leaf ip {
    type inet:ipv6-address-no-zone;
    status deprecated;
    description
      "Адрес IPv6 на интерфейсе.";
  }
  leaf prefix-length {
    type uint8 {
      range "0..128";
    }
    mandatory true;
    status deprecated;
    description
      "Размер префикса подсети.";
  }
}

```

```
}
leaf origin {
  type ip-address-origin;
  status deprecated;
  description
    "Источник этого адреса.";
}
leaf status {
  type enumeration {
    enum preferred {
      description
        "Это действительный адрес, который может указывать
        отправителя или получателя пакета.";
    }
    enum deprecated {
      description
        "Это действительный, но устаревший адрес, который не
        следует применять для отправителя в новых пакетах,
        но следует корректно обрабатывать в получаемых.";
    }
    enum invalid {
      description
        "Это недействительный адрес и его не следует
        указывать в поле отправителя или получателя пакета";
    }

    enum inaccessible {
      description
        "Адрес недоступен, поскольку интерфейс, в котором он
        связан, не работает.";
    }
    enum unknown {
      description
        "Статус невозможно определить по какой-то причине.";
    }
    enum tentative {
      description
        "Уникальность адреса на канале проверяется. Адреса с
        таким статусом не следует применять для общих
        коммуникаций, оставляя его лишь для проверки
        уникальности адреса.";
    }
    enum duplicate {
      description
        "Адрес сочтён неуникальным на канале и его
        недопустимо использовать.";
    }
    enum optimistic {
      description
        "Адрес доступен для использования с учётом
        ограничений, уникальность на канале проверяется.";
    }
  }
  status deprecated;
  description
    "Статус адреса. Большинство состояний соответствует
    протоколу IPv6 Stateless Address Autoconfiguration.";
  reference
    "RFC 4293: Management Information Base for the
    Internet Protocol (IP)
    - IpAddressStatusTC
    RFC 4862: IPv6 Stateless Address Autoconfiguration";
}
}

list neighbor {
  key "ip";
  status deprecated;
  description
    "Список сопоставлений адресов IPv4 с адресами канального
    уровня. Этот список представляет Neighbor Cache.";
  reference
    "RFC 4861: Neighbor Discovery for IP version 6 (IPv6)";

  leaf ip {
    type inet:ipv6-address-no-zone;
    status deprecated;
    description
      "Адрес IPv6 у соседнего узла.";
  }
  leaf link-layer-address {
    type yang:phys-address;
    status deprecated;
    description
      "Адрес канального уровня у соседнего узла.";
  }
}
```


Многие модели данных, определённые в этом модуле YANG обеспечивают возможность записи, создания и удаления (т. е. по умолчанию установлено `config true`). Эти узлы данных могут содержать конфиденциальную (`sensitive`) информацию, а также быть уязвимыми в некоторых сетевых средах. Операции записи (например, `edit-config`) для таких узлов без подобающей защиты могут оказывать нежелательное воздействие на работу сети. Ниже перечислены ветви (`subtree`) и узлы данных, которые могут включать конфиденциальную информацию или уязвимости.

ipv4/enabled u ipv6/enabled

Эти листья используются для включения или отключения IPv4 и IPv6 на конкретных интерфейсах. Включив протокол на интерфейсе, атакующий может получить возможность создания незащищённого пути к узлу (или через него, если включена маршрутизация). Отключив протокол на интерфейсе, атакующий сможет вынудить к маршрутизации пакетов через другой интерфейс или заблокировать доступ к части или всей сети по этому протоколу.

ipv4/address u ipv6/address

Эти списки задают адреса IP, установленные на интерфейсе. Изменяя эту информацию, атакующий сможет заставить узел игнорировать направленные ему сообщения или воспринимать (по меньшей мере на уровне IP) сообщения, которое он иначе бы игнорировал. Использование фильтрации или защищённых связей может снизить уровень возможных повреждений в последнем случае.

ipv4/forwarding u ipv6/forwarding

Эти листья позволяют клиенту включать и выключать функции пересылки. Запретив пересылку, атакующий может потенциально заблокировать обслуживание пользователей. Включив функции пересылки, атакующий сможет открыть для себя проход, что может привести к нежелательному транзиту пакетов или проникновению злоумышленника в сеть в обход механизмов защиты.

ipv6/autoconf

Листья в этой ветви управляют автоматической настройкой адресов IPv6 и, в частности, возможностью использования временных адресов. Изменяя соответствующие листья, атакующий может влиять на адреса, применяемые узлом и, опосредованно, на приватность пользователей узла.

ipv4/mtu u ipv6/mtu

Установка для этих листьев очень малых значений может существенно замедлить работу интерфейсов.

7. Литература

7.1. Нормативные документы

- [RFC791] Postel, J., "Internet Protocol", STD 5, [RFC 791](#), DOI 10.17487/RFC0791, September 1981, <<https://www.rfc-editor.org/info/rfc791>>.
- [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>>.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", [RFC 4861](#), DOI 10.17487/RFC4861, September 2007, <<https://www.rfc-editor.org/info/rfc4861>>.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <<https://www.rfc-editor.org/info/rfc4862>>.
- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, DOI 10.17487/RFC4941, September 2007, <<https://www.rfc-editor.org/info/rfc4941>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", [RFC 5246](#), DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [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>>.
- [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>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, [RFC 8200](#), DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [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>>.
- [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>>.
- [W3C.REC-xml-20081126] Bray, T., Paoli, J., Sperberg-McQueen, M., Maler, E., and F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", World Wide Web Consortium Recommendation REC-xml-20081126, November 2008, <<https://www.w3.org/TR/2008/REC-xml-20081126>>.

7.2. Дополнительная литература

- [RFC826] Plummer, D., "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48-bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, [RFC 826](#), DOI 10.17487/RFC826, November 1982, <<https://www.rfc-editor.org/info/rfc826>>.
- [RFC4293] Routhier, S., Ed., "Management Information Base for the Internet Protocol (IP)", [RFC 4293](#), DOI 10.17487/RFC4293, April 2006, <<https://www.rfc-editor.org/info/rfc4293>>.
- [RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", [RFC 7217](#), DOI 10.17487/RFC7217, April 2014, <<https://www.rfc-editor.org/info/rfc7217>>.
- [RFC8022] Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", [RFC 8022](#), DOI 10.17487/RFC8022, November 2016, <<https://www.rfc-editor.org/info/rfc8022>>.
- [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>>.

Приложение А. Пример отклика NETCONF <get-config>

В этом приложении дан пример отклика на запрос NETCONF <get-config> для рабочего хранилища конфигурации на устройстве, которое реализует определенную в этом документе модель данных.

Фрагменты XML [W3C.REC-xml-20081126] в этом и следующем приложении даны лишь в качестве примеров.

```
<rpc-reply
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
  message-id="101">
  <data>
    <interfaces
      xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
      xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
      <interface>
        <name>eth0</name>
        <type>ianaift:ethernetCsmacd</type>
        <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
          <address>
            <ip>192.0.2.1</ip>
            <prefix-length>24</prefix-length>
          </address>
        </ipv4>
        <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
          <mtu>1280</mtu>
          <address>
            <ip>2001:db8::10</ip>
            <prefix-length>32</prefix-length>
          </address>
          <dup-addr-detect-transmits>0</dup-addr-detect-transmits>
        </ipv6>
      </interface>
    </interfaces>
  </data>
</rpc-reply>
```

Приложение В. Пример отклика NETCONF <get-data>

В этом приложении дан пример отклика на запрос NETCONF <get-data> для хранилища состояния на устройстве, поддерживающем описанную в этом документе модель данных.

В примере используется аннотация origin, определенная в модуле ietf-origin [RFC8342].

```
<rpc-reply
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
  message-id="101">
  <data xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-datastores">
    <interfaces
      xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
      xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type"
      xmlns:or="urn:ietf:params:xml:ns:yang:ietf-origin">
      <interface or:origin="or:intended">
        <name>eth0</name>
        <type>ianaift:ethernetCsmacd</type>
        <!-- other parameters from ietf-interfaces omitted -->
```

```
<ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
  <enabled or:origin="or:default">true</enabled>
  <forwarding or:origin="or:default">false</forwarding>
  <mtu or:origin="or:system">1500</mtu>
  <address>
    <ip>192.0.2.1</ip>
    <prefix-length>24</prefix-length>
    <origin>static</origin>
  </address>
  <neighbor or:origin="or:learned">
    <ip>192.0.2.2</ip>
    <link-layer-address>
      00:00:5E:00:53:AB
    </link-layer-address>
  </neighbor>
</ipv4>
<ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
  <enabled or:origin="or:default">true</enabled>
  <forwarding or:origin="or:default">false</forwarding>
  <mtu>1280</mtu>
  <address>
    <ip>2001:db8::10</ip>
    <prefix-length>32</prefix-length>
    <origin>static</origin>
    <status>preferred</status>
  </address>
  <address or:origin="or:learned">
    <ip>2001:db8::1:100</ip>
    <prefix-length>32</prefix-length>
    <origin>dhcp</origin>
    <status>preferred</status>
  </address>
  <dup-addr-detect-transmits>0</dup-addr-detect-transmits>
  <neighbor or:origin="or:learned">
    <ip>2001:db8::1</ip>
    <link-layer-address>
      00:00:5E:00:53:AB
    </link-layer-address>
    <origin>dynamic</origin>
    <is-router/>
    <state>reachable</state>
  </neighbor>
  <neighbor or:origin="or:learned">
    <ip>2001:db8::4</ip>
    <origin>dynamic</origin>
    <state>incomplete</state>
  </neighbor>
</ipv6>
</interface>
</interfaces>
</data>
</rpc-reply>
```

Благодарности

Автор благодарит Jeffrey Lange, Ladislav Lhotka, Juergen Schoenwaelder и Dave Thaler за их полезные комментарии.

Адрес автора

Martin Bjorklund

Tail-f Systems

Email: mbj@tail-f.com

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru