Internet Engineering Task Force (IETF)
Request for Comments: 8348
Category: Standards Track
ISSN: 2070-1721

A. Bierman
YumaWorks
M. Bjorklund
Tail-f Systems
J. Dong
Huawei Technologies
D. Romascanu

March 2018

# A YANG Data Model for Hardware Management

Модель данных YANG для управления оборудованием

**Аннотация**

Этот документ определяет модель данных YANG для управления оборудованием на одном сервере.

**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/rfc8348.



# Оглавление



## 1. Введение

Этот документ определяет модель данных YANG [RFC7950] для управления оборудованием на одном сервере.

Модель данных включает конфигурацию и состояние системы (сведения о статусе и счётчики статистики).

Модель YANG в этом документе соответствует архитектуре хранилищ сетевой информации NMDA, определённой в [RFC8342]. Для реализаций, ещё не поддерживающих NMDA, в Приложении A задан временный модуль, включающий лишь данные состояния системы.

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

## 1.1. 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.

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

- client - клиент;
- server - сервер;
- configuration - конфигурация;
- system state - состояние системы;
- operational state - рабочее состояние;
- intended configuration - предполагаемая (предусмотренная) конфигурация.

## 1.2. Диаграммы деревьев

В документе применяется графическое представление моделей данных, описанное в [RFC8340].

# *2. Цели*

В этом разделе указаны некоторые цели проектирования модели данных для оборудования.

- Модель должна поддерживать многие общие свойства, служащие для идентификации аппаратных компонентов.
- Важные сведения и состояния для компонентов оборудования требуется собирать с устройств, поддерживающих модель данных оборудования.
- Модель данных следует подходить в неизменном виде для новых реализаций.
- Заданную в этом документе модель данных можно реализовать на системах, которые реализуют ENTITY-MIB, поэтому нужно чёткое сопоставления между моделью данных оборудования и ENTITY-MIB.
- Модели данных следует поддерживать предварительную подготовку аппаратных компонентов.

## 3. Модель данных оборудования

Этот документ определяет модуль YANG ietf-hardware, структура которого показана ниже.

```
module: ietf-hardware
  +--rw hardware
     +--ro last-change?   yang:date-and-time
     +--rw component* [name]
        +--rw name               string
        +--rw class              identityref
        +--ro physical-index?    int32 {entity-mib}?
        +--ro description?       string
        +--rw parent?            -> ../../component/name
        +--rw parent-rel-pos?    int32
        +--ro contains-child*    -> ../../component/name
        +--ro hardware-rev?      string
        +--ro firmware-rev?      string
        +--ro software-rev?      string
        +--ro serial-num?        string
        +--ro mfg-name?          string
        +--ro model-name?        string
        +--rw alias?             string
        +--rw asset-id?          string
        +--ro is-fru?            boolean
        +--ro mfg-date?          yang:date-and-time
        +--rw uri*               inet:uri
        +--ro uuid?              yang:uuid
        +--rw state {hardware-state}?
        |  +--ro state-last-changed?   yang:date-and-time
        |  +--rw admin-state?          admin-state
        |  +--ro oper-state?           oper-state
        |  +--ro usage-state?          usage-state
        |  +--ro alarm-state?          alarm-state
        |  +--ro standby-state?        standby-state
        +--ro sensor-data {hardware-sensor}?
           +--ro value?               sensor-value
           +--ro value-type?          sensor-value-type
           +--ro value-scale?         sensor-value-scale
           +--ro value-precision?     sensor-value-precision
           +--ro oper-status?         sensor-status
           +--ro units-display?       string
           +--ro value-timestamp?     yang:date-and-time
           +--ro value-update-rate?   uint32

  notifications:
```

```
      +---n hardware-state-change
      +---n hardware-state-oper-enabled {hardware-state}?
      |  +--ro name?          -> /hardware/component/name
      |  +--ro admin-state?   -> /hardware/component/state/admin-state
      |  +--ro alarm-state?   -> /hardware/component/state/alarm-state
      +---n hardware-state-oper-disabled {hardware-state}?
         +--ro name?          -> /hardware/component/name
         +--ro admin-state?   -> /hardware/component/state/admin-state
         +--ro alarm-state?   -> /hardware/component/state/alarm-state
```

### 3.1. Списки компонентов

Представленная в документе модель данных для оборудования использует плоский список компонентов, каждый из которых указывается именем и имеет обязательный лист class.

Модуль iana-hardware определяет идентификаторы YANG для типов оборудования из поддерживаемого IANA реестра IANA-ENTITY-MIB.

Лист class служит идентификатором YANG, описывающим тип оборудования. Производителям рекомендуется напрямую использовать заданные IANA идентификаторы или выводить из них более конкретные обозначения.

## *4. Связь с ENTITY-MIB*

Если устройство реализует ENTITY-MIB [RFC6933], каждая запись в списке /hardware/component рабочего состояния отображается на EntPhysicalEntry. Объекты, доступные для записи в MIB отображаются в узлы config true в списке /hardware/component, за исключением entPhysicalSerialNum (открыт для записи в MIB) имеющего config false в модуле YANG.

Лист physical-index **должен** содержать значение, соответствующее entPhysicalIndex в entPhysicalEntry.

Лист class сопоставляется с entPhysicalClass и entPhysicalVendorType. Если значением class является идентификатор, производный от идентификатора в iana-hardware, либо просто идентификатор из iana-hardware, то entPhysicalClass содержит соответствующее перечисляемое значение IANAPhysicalClass. В иных случаях entPhysicalClass содержит IANAPhysicalClass other(1). Производителям рекомендуется задавать идентификатор (выведенный из идентификатора iana-hardware, если это возможно) для каждого своего (enterprise-specific) идентификатора регистрации, применяемого в entPhysicalVendorType, и использовать этот идентификатор для листа class.

В таблице 1 указаны узлы данных YANG и соответствующие объекты ENTITY-MIB.

*Таблица 1. Узлы модели данных YANG и связанные объекты ENTITY-MIB.*

| ***Узел данных YANG в /hardware/component*** | ***Объект ENTITY-MIB*** |
|---|---|
| name | entPhysicalName |
| class | entPhysicalClass |
| | entPhysicalVendorType |
| physical-index | entPhysicalIndex |
| description | entPhysicalDescr |
| parent | entPhysicalContainedIn |
| parent-rel-pos | entPhysicalParentRelPos |
| contains-child | entPhysicalChildIndex |
| hardware-rev | entPhysicalHardwareRev |
| firmware-rev | entPhysicalFirmwareRev |
| software-rev | entPhysicalSoftwareRev |
| serial-num | entPhysicalSerialNum |
| mfg-name | entPhysicalMfgName |
| model-name | entPhysicalModelName |
| alias | entPhysicalAlias |
| asset-id | entPhysicalAssetID |
| is-fru | entPhysicalIsFRU |
| mfg-date | entPhysicalMfgDate |
| uri | entPhysicalUris |
| uuid | entPhysicalUUID |

## *5. Связь с ENTITY-SENSOR-MIB*

Если устройство реализует ENTITY-SENSOR-MIB [RFC3433], каждая записи в списке /hardware/component, имеющая контейнер sensor-data, отображается на EntPhySensorEntry.

В таблице 2 указаны узлы данных YANG и соответствующие объекты ENTITY-SENSOR-MIB.

*Таблица 2. Узлы модели данных YANG и связанные объекты ENTITY-SENSOR-MIB.*

| ***Узел данных в YANG in/hardware/component/sensor-data*** | ***Объект ENTITY-SENSOR-MIB*** |
|---|---|
| value | entPhySensorValue |
| value-type | entPhySensorType |
| value-scale | entPhySensorScale |
| value-precision | entPhySensorPrecision |
| oper-status | entPhySensorOperStatus |
| units-display | entPhySensorUnitsDisplay |
| value-timestamp | entPhySensorValueTimeStamp |
| value-update-rate | entPhySensorValueUpdateRate |

## *6. Связь с ENTITY-STATE-MIB*

Если устройство реализует ENTITY-STATE-MIB [RFC4268], каждая запись в списке /hardware/component, имеющая контейнер state, отображается на EntStateEntry.

В таблице 3 указаны узлы данных YANG и соответствующие объекты ENTITY-STATE-MIB.

*Таблица 3. Узлы модели данных YANG и связанные объекты ENTITY-STATE-MIB.*

| ***Узел данных в YANG /hardware/component/state*** | ***Объект ENTITY-STATE-MIB*** |
|---|---|
| state-last-changed | entStateLastChanged |
| admin-state | entStateAdmin |
| oper-state | entStateOper |
| usage-state | entStateUsage |
| alarm-state | entStateAlarm |
| standby-state | entStateStandby |

# *7. Модули YANG для оборудования*

## 7.1. Модуль ietf-hardware

Этот модуль YANG импортирует определения типов (typedef) из [RFC6991].

```
<CODE BEGINS> file "ietf-hardware@2018-03-13.yang"

  module ietf-hardware {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-hardware";
    prefix hw;

    import ietf-inet-types {
      prefix inet;
    }
    import ietf-yang-types {
      prefix yang;
    }
    import iana-hardware {
      prefix ianahw;
    }

    organization
      "IETF NETMOD (Network Modeling) Working Group";

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

       Editor:   Andy Bierman
                 <mailto:andy@yumaworks.com>

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

       Editor:   Jie Dong
                 <mailto:jie.dong@huawei.com>

       Editor:   Dan Romascanu
                 <mailto:dromasca@gmail.com>";

    description
      "Модуль содержит определения YANG для управления оборудованием.

       Модель данных разработана для архитектуры NMDA,
       определённой в 8342.

       Авторские права (Copyright (c) 2018) 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 8348, где
       правовые вопросы рассмотрены более полно.";

    revision 2018-03-13 {
      description
        "Исходный выпуск.";
      reference
        "RFC 8348: A YANG Data Model for Hardware Management";
    }

    /*
     * Свойства (функции)
     */
    feature entity-mib {
      description
        "Указывает, что устройство реализует ENTITY-MIB.";
      reference
```

```
      "RFC 6933: Entity MIB (Version 4)";
  }

  feature hardware-state {
    description
      "Указывает поддержку объектов ENTITY-STATE-MIB.";
    reference
      "RFC 4268: Entity State MIB";
  }

  feature hardware-sensor {
    description
      "Указывает поддержку объектов ENTITY-SENSOR-MIB.";
    reference
      "RFC 3433: Entity Sensor Management Information Base";
  }

  /*
   * Определения типов
   */
  typedef admin-state {
    type enumeration {
      enum unknown {
        value 1;
        description
          "Ресурс не способен сообщить административный статус.";
      }
      enum locked {
        value 2;
        description
          "Использование ресурса запрещено административно.";
      }
      enum shutting-down {
        value 3;
        description
          "Использование ресурса ограничено текущими экземплярами.";
      }
      enum unlocked {
        value 4;
        description
          "Использование ресурса не запрещено административно.";
      }
    }
    description
      "Возможные административные состояния.";
    reference
      "RFC 4268: Entity State MIB - EntityAdminState";
  }

  typedef oper-state {
    type enumeration {
      enum unknown {
        value 1;
        description
          "Ресурс не способен сообщить рабочее состояние.";
      }
      enum disabled {
        value 2;
        description
          "Ресурс неработоспособен.";
      }
      enum enabled {
        value 3;
        description
          "Ресурс частично или полностью работоспособен.";
      }
      enum testing {
        value 4;
        description
          "Ресурс в настоящее время тестируется, поэтому не
           способен сообщить своё рабочее состояние.";
      }
    }
    description
      "Возможные значения рабочего состояния.";
    reference
      "RFC 4268: Entity State MIB - EntityOperState";
  }

  typedef usage-state {
    type enumeration {
      enum unknown {
        value 1;
        description
          "Ресурс не способен сообщить состояние использования.";
      }
```

```
        enum idle {
          value 2;
          description
            "Ресурс не обслуживает пользователей.";
        }
        enum active {
          value 3;
          description
            "Ресурс используется и имеет достаточно возможностей для
             предоставления дополнительным пользователям.";
        }
        enum busy {
          value 4;
          description
            "Ресурс используется, но не имеет возможностей для
             предоставления дополнительным пользователям.";
        }
      }
      description
        "Возможные значения состояния использования.";
      reference
        "RFC 4268: Entity State MIB -  EntityUsageState";
    }

    typedef alarm-state {
      type bits {
        bit unknown {
          position 0;
          description
            "Ресурс не способен сообщить статус состояния тревоги.";
        }
        bit under-repair {
          position 1;
          description
            "Ресурс ремонтируется и этот бит (в зависимости от
             реализации) может лишать смысла другие биты строки.";
        }
        bit critical {
          position 2;
          description
            "Для ресурса имеется хотя бы один критический
             сигнал тревоги.";
        }
        bit major {
          position 3;
          description
            "Для ресурса имеется хотя бы один важный (major)
             сигнал тревоги.";
        }
        bit minor {
          position 4;
          description
            "Для ресурса имеется хотя бы один маловажный (minor)
             сигнал тревоги.";
        }
        bit warning {
          position 5;
          description
            "Для ресурса имеется хотя бы одно предупреждение.";
        }
        bit indeterminate {
          position 6;
          description
            "Для ресурса имеется хотя бы один сигнал тревоги,
             важность которого невозможно определить.";
        }
      }
      description
        "Возможные значения сигналов тревоги. Сигналом служит
         сохраняющаяся индикация ошибки или предупреждения.

         Когда в этом атрибуте не установлен ни один бит, активных
         сигналов тревоги нет, и ресурс не находится в ремонте.";
      reference
        "RFC 4268: Entity State MIB - EntityAlarmStatus";
    }

    typedef standby-state {
      type enumeration {
        enum unknown {
          value 1;
          description
            "Ресурс не способен сообщить статус ожидания (standby).";
        }
        enum hot-standby {
          value 2;
```

```
        description
          "Ресурс не предоставляет услуг, но сразу может принять роль
           резервного ресурса без необходимости инициализации и будет
           содержать ту же информацию, которая была в резервируемом.";
      }
      enum cold-standby {
        value 3;
        description
          "Ресурс является резервным для другого ресурса, но не сможет
           сразу принять роль резервируемого и требует инициализации.";
      }
      enum providing-service {
        value 4;
        description
          "Ресурс предоставляет услуги.";
      }
    }
    description
      "Возможные значения статуса ожидания (standby).";
    reference
      "RFC 4268: Entity State MIB - EntityStandbyStatus";
  }

  typedef sensor-value-type {
    type enumeration {
      enum other {
        value 1;
        description
          "Измерения (меры), отличные от указанных ниже.";
      }
      enum unknown {
        value 2;
        description
          "Неизвестная мера или произвольные относительные числа";
      }
      enum volts-AC {
        value 3;
        description
          "Мера электрического потенциала (переменный ток).";
      }
      enum volts-DC {
        value 4;
        description
          "Мера электрического потенциала (постоянный ток).";
      }
      enum amperes {
        value 5;
        description
          "Мера электрического тока.";
      }
      enum watts {
        value 6;
        description
          "Мера мощности.";
      }
      enum hertz {
        value 7;
        description
          "Мера частоты.";
      }
      enum celsius {
        value 8;
        description
          "Мера температуры.";
      }
      enum percent-RH {
        value 9;
        description
          "Мера относительной влажности (%).";
      }
      enum rpm {
        value 10;
        description
          "Число оборотов в минуту.";
      }
      enum cmm {
        value 11;
        description
          "Число кубометров в минуту (воздушный поток).";
      }
      enum truth-value {
        value 12;
        description
          "Логическое значение - 1 (true) или 2 (false)";
      }
    }
```

```
      description
        "Узел этого типа представляет физическое значение датчика.
         Фактическая величина определяется этим узлом и связанным узлом
         коэффициента измерения sensor-value-scale.

         Узлы этого типа СЛЕДУЕТ определять вместе с узлами типа
         sensor-value-scale и sensor-value-precision. Эти три типа служат
         для указания семантики узла типа sensor-value.";
      reference
        "RFC 3433: Entity Sensor Management Information Base -
                   EntitySensorDataType";
    }

    typedef sensor-value-scale {
      type enumeration {
        enum yocto {
          value 1;
          description
            "Коэффициент измерения 10^-24.";
        }
        enum zepto {
          value 2;
          description
            "Коэффициент измерения 10^-21.";
        }
        enum atto {
          value 3;
          description
            "Коэффициент измерения 10^-18.";
        }
        enum femto {
          value 4;
          description
            "Коэффициент измерения 10^-15.";
        }
        enum pico {
          value 5;
          description
            "Коэффициент измерения 10^-12.";
        }
        enum nano {
          value 6;
          description
            "Коэффициент измерения 10^-9.";
        }
        enum micro {
          value 7;
          description
            "Коэффициент измерения 10^-6.";
        }
        enum milli {
          value 8;
          description
            "Коэффициент измерения 10^-3.";
        }
        enum units {
          value 9;
          description
            "Коэффициент измерения 10^0.";
        }
        enum kilo {
          value 10;
          description
            "Коэффициент измерения 10^3.";
        }
        enum mega {
          value 11;
          description
            "Коэффициент измерения 10^6.";
        }
        enum giga {
          value 12;
          description
            "Коэффициент измерения 10^9.";
        }
        enum tera {
          value 13;
          description
            "Коэффициент измерения 10^12.";
        }
        enum peta {
          value 14;
          description
            "Коэффициент измерения 10^15.";
        }
        enum exa {
```

```
        value 15;
        description
          "Коэффициент измерения 10^18.";
      }
      enum zetta {
        value 16;
        description
          "Коэффициент измерения 10^21.";
      }
      enum yotta {
        value 17;
        description
          "Коэффициент измерения 10^24.";
      }
    }
    description
      "Узел этого типа представляет коэффициент измерения, для
       единиц международной системы СИ. Фактическое значение
       определяется этим узлом и соответствующим sensor-value-type.

       Узлы этого типа СЛЕДУЕТ определять вместе с узлами типа
       sensor-value-type и sensor-value-precision. Эти три типа служат
       для указания семантики узла типа sensor-value.";
    reference
      "RFC 3433: Entity Sensor Management Information Base -
                 EntitySensorDataScale";
  }

  typedef sensor-value-precision {
    type int8 {
      range "-8 .. 9";
    }
    description
      "Узел этого типа представляет точность значения датчика.

       Узлы этого типа СЛЕДУЕТ определять вместе с узлами типа
       sensor-value-type и sensor-value-scale. Эти три типа служат
       для указания семантики узла типа sensor-value.";

       Значения от 1 до 9, указывающее число десятичных цифр дробной
       части, соответствующего значения sensor-value с фиксированной
       точкой (запятой). Значение от -8 до -1 указывают число значимых
       (точных) цифр соответствующего значения sensor-value с
       фиксированной точкой. Значение 0 указывает, что значение
       sensor-value не является числом с фиксированной точкой.

       Разработчики серверов должны выбирать значение для узла
       sensor-value-precision так, чтобы точность и значащие цифры
       связанного значения sensor-value указывались корректно.
       Например, узел, представляющий датчик температуры, способный
       измерять значения от 0 до 100 C с шагом 0,1 C и точностью
       +/- 0,05 C, будет иметь sensor-value-precision 1,
       sensor-value-scale units и sensor-value от 0 до 1000.
       Значение sensor-value будет интерпретироваться как C * 10.";
    reference
      "RFC 3433: Entity Sensor Management Information Base -
                 EntitySensorPrecision";
  }

  typedef sensor-value {
    type int32 {
      range "-1000000000 .. 1000000000";
    }
    description
     "Узел этого типа представляет значение датчика.

      Узлы этого типа СЛЕДУЕТ определять вместе с узлами типа
      sensor-value-type, sensor-value-scale и sensor-value-precision.
      Эти три типа служат для указания семантики узла данного типа.";

      Если связанный узел sensor-value-type имеет значение voltsAC,
      voltsDC, amperes, watts, hertz, celsius или cmm, узел этого типа
      ДОЛЖЕН содержать значение с фиксированной точкой (запятой) из
      диапазона -999999999 - 999999999. Значение -1000000000 указывает
      исчерпание (underflow), 1000000000 — переполнение (overflow).
      Значение sensor-value-precision указывает число цифр дробной
      части, представленных в связанном узле sensor-value.

      Если связанный узел sensor-value-type имеет значение percentRH,
      этот узел ДОЛЖЕН содержать число от 0 до 100.

      Если связанный узел sensor-value-type имеет значение rpm,
      этот узел ДОЛЖЕН содержать число от -999999999 до +999999999.

      Если связанный узел sensor-value-type имеет значение
      truth-value, этот узел ДОЛЖЕН содержать 1 (true) или 2 (false).
```

```
      Если связанный узел sensor-value-type имеет значение other или
      unknown,  этот узел ДОЛЖЕН содержать число от -1000000000 до
      1000000000.";
    reference
      "RFC 3433: Entity Sensor Management Information Base -
                 EntitySensorValue";
  }

  typedef sensor-status {
    type enumeration {
      enum ok {
        value 1;
        description
          "Указывает возможность получить значение датчика.";
      }
      enum unavailable {
        value 2;
        description
          "Указывает, что сервер сейчас не может получить значение
           датчика.";
      }
      enum nonoperational {
        value 3;
        description
          "Указывает, что сервер считает датчик неисправным. Это может
           быть аппаратный отказ (обрыв провода) или мягкая причина
           вроде выхода за пределы диапазона, дрожание или скачки.";
      }
    }
    description
      "Узел этого типа представляет рабочее состояние физического
       датчика.";
    reference
      "RFC 3433: Entity Sensor Management Information Base -
                 EntitySensorStatus";
  }

  /*
   * Узлы данных
   */
  container hardware {
    description
      "Узлы, представляющие компоненты. Если сервер поддерживает
       настройку аппаратных компонентов, эта модель данных создаётся
       в хранилище конфигурации, поддерживаемом сервером. Эти
       сведения обеспечивает leaf-list datastore для модуля
       ietf-hardware в библиотеке YANG.";

    leaf last-change {
      type yang:date-and-time;
      config false;
      description
        "Время изменения списка /hardware/component в рабочем
         состоянии.";
    }

    list component {
      key name;
      description
        "Список компонентов. Когда сервер обнаруживает новый
         аппаратный компонент, он инициализирует запись списка в
         рабочем состоянии. Если сервер не поддерживает настройку
         аппаратных компонентов, записи списка в рабочем состоянии
         инициализируются значениями для всех узлов, найденных
         реализацией. В остальных случаях выполняются указанные
         ниже процедуры.
           1. Если в списке /hardware/component предполагаемой
              (intended) конфигурации есть запись со значениями узлов
              class, parent и parent-rel-pos, равными обнаруженным,
              запись списка в рабочем состоянии инициализируется
              настроенными значениями, включая name.
           2. В ином случае (нет соответствующей записи конфигурации)
              записи списка в рабочем состоянии инициализируется
              значениями для всех узлов, найденных реализацией.

         Если список /hardware/component в предполагаемой конфигурации
         изменён, система ДОЛЖНА вести себя как при реинициализации
         и следовать процедуре п. (1).";
      reference
        "RFC 6933: Entity MIB (Version 4) - entPhysicalEntry";

      leaf name {
        type string;
        description
          "Имя компонента (может отличаться от entPhysicalName).";
```

```
      }

      leaf class {
        type identityref {
          base ianahw:hardware-class;
        }
        mandatory true;
        description
          "Оборудование общего назначения.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalClass";
      }

      leaf physical-index {
        if-feature entity-mib;
        type int32 {
          range "1..2147483647";
        }
        config false;
        description
          "Представляет entPhysicalIndex для entPhysicalEntry.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalIndex";
      }

      leaf description {
        type string;
        config false;
        description
          "Текстовое описание компонента. Узлу следует содержать
           строку, указывающую производителя компонента и следует
           указывать разные значения для каждой версии или модели.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalDescr";
      }

      leaf parent {
        type leafref {
          path "../../component/name";
          require-instance false;
        }
        description
          "Имя компонента, физически включающего этот компонент. Если
           этот лист не создаётся, это указывает, что компонент не
           содержится в каком-либо ином компоненте.

           Если физический компонент входит в несколько физических
           компонентов (например, модуль двойной ширины), узел
           содержит имя одного из этих компонентов. Реализация ДОЛЖНА
           использовать одно имя для каждого экземпляра узла.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalContainedIn";
      }

      leaf parent-rel-pos {
        type int32 {
          range "0 .. 2147483647";
        }
        description
          "Указывает относительное положение дочернего компонента
           среди братских, к каковым относятся компоненты, имеющие
           одно значение узла parent и одно базовое отождествление
           для узла class. Отметим, что второе правило обеспечивает
           реализациям гибкость нумерации компонентов. Например,
           некоторые реализации могут иметь одну серию номеров для
           всех компонентов, производных от ianahw:port, а другие -
           применять для них разные серии номеров (скажем, одну для
           разъемов RJ45, другую для SFP).";

        reference
          "RFC 6933: Entity MIB (Version 4) -
                     entPhysicalParentRelPos";
      }

      leaf-list contains-child {
        type leafref {
          path "../../component/name";
        }
        config false;
        description
          "Имя содержащегося компонента.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalChildIndex";
      }

      leaf hardware-rev {
```

```
        type string;
        config false;
        description
          "Фирменная строка аппаратного выпуска компонента.
           Предпочтительно указывать идентификатор, помещаемый
           производителем на сам компонент (при наличии).";
        reference
          "RFC 6933: Entity MIB (Version 4) -
                     entPhysicalHardwareRev";
      }

      leaf firmware-rev {
        type string;
        config false;
        description
          "Фирменная строка выпуска микрокода для компонента.";
        reference
          "RFC 6933: Entity MIB (Version 4) -
                     entPhysicalFirmwareRev";
      }

      leaf software-rev {
        type string;
        config false;
        description
          "Фирменная строка программного выпуска компонента.";
        reference
          "RFC 6933: Entity MIB (Version 4) -
                     entPhysicalSoftwareRev";
      }

      leaf serial-num {
        type string;
        config false;
        description
          "Фирменная строка серийного номера компонента.
           Предпочтительно указывать номер, помещаемый
           производителем на сам компонент (при наличии).";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalSerialNum";
      }

      leaf mfg-name {
        type string;
        config false;
        description
          "Название изготовителя для физического компонента.
           Предпочтительно указывать название, помещаемое
           изготовителем на сам компонент (при наличии).";

           Отметим, что сравнение значений model-name, firmware-rev,
           software-rev и serial-num имеет смысл лишь для
           компонентов с одним значением mfg-name.

           Если заданное изготовителем название физического
           компонента неизвестно серверу, этот узел не создаётся.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalMfgName";
      }

      leaf model-name {
        type string;
        config false;
        description
          "Фирменная строка идентификатора модели физического
           компонента. Предпочтительно указывать видимый клиенту
           код, напечатанный на самом компоненте (при наличии).

           Если заданный производителем код модели физического
           компонента неизвестно серверу, этот узел не создаётся.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalModelName";
      }

      leaf alias {
        type string;
        description
          "Псевдоним имени компонента, заданный администратором сети,
           сохраняющийся при выключении или перезагрузке компонента.

           Если псевдоним не задан, сервер МОЖЕТ установить для узла
           локально уникальное значение в рабочем состоянии.

           Реализация сервера МОЖЕТ сопоставить этот лист с объектом
           MIB entPhysicalAlias. Такой реализации нужен механизм для
           обработки различий в размере и разрешённых символах между
```

```
        этим листом и entPhysicalAlias. Задание такого механизма
        выходит за рамки этого документа.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalAlias";
  }

  leaf asset-id {
    type string;
    description
      "Заданный пользователем идентификатор отслеживания активов
       для компонента.

       Реализация сервера МОЖЕТ сопоставить этот лист с объектом
       MIB entPhysicalAssetID. Такой реализации нужен механизм для
       обработки различий в размере и разрешённых символах между
       этим листом и entPhysicalAssetID. Задание такого механизма
       выходит за рамки этого документа.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalAssetID";
  }

  leaf is-fru {
    type boolean;
    config false;
    description
      "Указывает, считает ли производитель этот компонент
       заменяемым в полевых условиях. Значение true указывает
       возможность такой замены. Для всех компонентов, постоянно
       содержащихся в этом компоненте для этого узла следует
       возвращать значение false.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalIsFRU";
  }

  leaf mfg-date {
    type yang:date-and-time;
    config false;
    description
      "Дата изготовления компонента.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalMfgDate";
  }

  leaf-list uri {
    type inet:uri;
    description
      "Идентификационные сведения о компоненте.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalUris";
  }

  leaf uuid {
    type yang:uuid;
    config false;
    description
      "Глобально уникальный идентификатор (UUID) компонента.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalUUID";
  }

  container state {
    if-feature hardware-state;
    description
      "Относящиеся к состоянию узлы";
    reference
      "RFC 4268: Entity State MIB";

    leaf state-last-changed {
      type yang:date-and-time;
      config false;
      description
        "Дата и время смены любого из значений admin-state,
         oper-state, usage-state, alarm-state или
         standby-state для этого компонента.

         Если с момента последней реинициализации системы не было
         изменений, узел содержит дату и время инициализации
         локальной системы. Если не было изменений с момента
         установки компонента в локальную систему, указывается
         дата и время такой установки.";
      reference
        "RFC 4268: Entity State MIB - entStateLastChanged";
    }

    leaf admin-state {
      type admin-state;
```

```
        description
          "Административное состояние компонента. Указывает
           административное разрешение обслуживать другие компоненты
           в иерархии содержания, а также других пользователей этих
           услуг, заданных мерами, выходящими за рамки этого модуля.

           Некоторые компоненты раскрывают лишь часть остальных
           значений административного состояния. Некоторые
           компоненты нельзя блокировать, поэтому узел показывает
           лишь статус unlocked. Другие компоненты нельзя аккуратно
           отключить и узел не показывает статус shutting-down.";
        reference
          "RFC 4268: Entity State MIB - entStateAdmin";
      }

      leaf oper-state {
        type oper-state;
        config false;
        description
          "Рабочее состояние компонента. Отметим, что этот узел не
           за административным статусом, т. е. административный
           статус down не означает рабочего статуса disabled.

           Отметим, что некоторые реализации могут не поддерживать
           точное отображение oper-state, когда admin-state для узла
           отличается от unlocked. В таких случаях узел ДОЛЖЕН иметь
           значение unknown.";
        reference
          "RFC 4268: Entity State MIB - entStateOper";
      }

      leaf usage-state {
        type usage-state;
        config false;
        description
          "Состояние использования компонента, отражающее его
           способность обслуживать дополнительные компоненты в
           иерархии содержания.

           Некоторые компоненты раскрывают лишь часть значений
           статуса использования. Компоненты, не способные
           обслуживать какие-либо компоненты в иерархии размещения,
           всегда будут иметь статус busy. В некоторых случаях
           компонент может поддерживать лишь ещё один компонент в
           иерархии размещения и будет раскрывать лишь значение
           idle или busy.";
        reference
          "RFC 4268: Entity State MIB - entStateUsage";
      }

      leaf alarm-state {
        type alarm-state;
        config false;
        description
          "Состояние сигналов тревоги для компонента без учёта
           сигналов в дочерних компонентах иерархии размещения.";
        reference
          "RFC 4268: Entity State MIB - entStateAlarm";
      }

      leaf standby-state {
        type standby-state;
        config false;
        description
          "Состояние ожидания (standby) для компонента.

           Некоторые компоненты раскрывают лишь часть значений
           статуса ожидания. Компоненты, не способные работать
           в режиме ожидания будут иметь для этого узла значение
           providing-service.";
        reference
          "RFC 4268: Entity State MIB - entStateStandby";
      }
    }

    container sensor-data {
      when 'derived-from-or-self(../class,
                                 "ianahw:sensor")' {
        description
          "Узлы данных для компонентов типа sensor";
      }
      if-feature hardware-sensor;
      config false;

      description
        "Относящиеся к датчикам узлы.";
```

```
        reference
          "RFC 3433: Entity Sensor Management Information Base";

        leaf value {
          type sensor-value;
          description
            "Последнее измерение, полученное сервером от датчика.

             Клиенту, периодически просматривающему этот узел,
             следует извлекать и узлы value-type, value-scale и
             value-precision, поскольку они тоже могут изменяться.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorValue";
        }

        leaf value-type {
          type sensor-value-type;
          description
            "Тип единиц, связанных со значением датчика";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorType";
        }

        leaf value-scale {
          type sensor-value-scale;
          description
            "Коэффициент (степень 10) измерения для значения узла";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorScale";
        }

        leaf value-precision {
          type sensor-value-precision;
          description
            "Число десятичных знаков точности значения узла.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorPrecision";
        }

        leaf oper-status {
          type sensor-status;
          description
            "Рабочее состояние датчика.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorOperStatus";
        }

        leaf units-display {
          type string;
          description
            "Текстовое описание единиц, которые следует применять при
             выводе значения датчика.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorUnitsDisplay";
        }

        leaf value-timestamp {
          type yang:date-and-time;
          description
            "Время последнего извлечения сервером статуса или
             значения этого датчика.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorValueTimeStamp";
        }

        leaf value-update-rate {
          type uint32;
          units "milliseconds";
          description
            "Интервал, с которым сервер обновляет связанный узел
             value в миллисекундах. Значение 0 указывает, что:
              - значение датчика обновляется по запросу (например,
                при опросе сервером для get-request);
              - значение датчика обновляется при его изменении
                (по событию);
              - сервер не знает частоты обновления.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorValueUpdateRate";
```

```
          }
        }
      }
    }

    /*
     * Уведомления
     */
    notification hardware-state-change {
      description
        "Уведомление hardware-state-change создается при изменении
         значения /hardware/last-change в рабочем состоянии.";
      reference
        "RFC 6933: Entity MIB (Version 4) - entConfigChange";
    }

    notification hardware-state-oper-enabled {
      if-feature hardware-state;
      description
        "Уведомление hardware-state-oper-enabled указывает переход
         компонента в состояние enabled.";

      leaf name {
        type leafref {
          path "/hardware/component/name";
        }
        description
          "Имя компонента, перешедшего в состояние enabled.";
      }
      leaf admin-state {
        type leafref {
          path "/hardware/component/state/admin-state";
        }
        description
          "Административное состояние компонента.";
      }
      leaf alarm-state {
        type leafref {
          path "/hardware/component/state/alarm-state";
        }
        description
          "Статус сигналов тревоги для компонента.";
      }
      reference
        "RFC 4268: Entity State MIB - entStateOperEnabled";
    }

    notification hardware-state-oper-disabled {
      if-feature hardware-state;
      description
        "Уведомление hardware-state-oper-disabled указывает переход
         компонента в состояние disabled.";

      leaf name {
        type leafref {
          path "/hardware/component/name";
        }
        description
          "Имя компонента, перешедшего в состояние disabled.";
      }
      leaf admin-state {
        type leafref {
          path "/hardware/component/state/admin-state";
        }
        description
          "Административное состояние компонента.";
      }
      leaf alarm-state {
        type leafref {
          path "/hardware/component/state/alarm-state";
        }
        description
          "Статус сигналов тревоги для компонента.";
      }
      reference
        "RFC 4268: Entity State MIB - entStateOperDisabled";
    }
  }
  <CODE ENDS>
```

## 7.2. Модуль iana-hardware

```
<CODE BEGINS> file "iana-hardware@2018-03-13.yang"
   module iana-hardware {
```

```
yang-version 1.1;
namespace "urn:ietf:params:xml:ns:yang:iana-hardware";
prefix ianahw;

organization "IANA";
contact
  "        Internet Assigned Numbers Authority

   Postal: ICANN
           12025 Waterfront Drive, Suite 300
           Los Angeles, CA  90094-2536
           United States of America

   Tel:    +1 310 301 5800
   E-Mail: iana@iana.org>";

description
  "Заданные IANA идентификаторы классов оборудования.

   Последний выпуск этого модуля YANG доступен на сайте IANA.
   Запросы на выделение новых значений следует направлять в IANA
   по адресу iana@iana.org.

   Авторские права (Copyright (c) 2018) 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 8348, где
   правовые вопросы рассмотрены более полно.";
reference
  "https://www.iana.org/assignments/yang-parameters";

revision 2018-03-13 {
  description
    "Исходный выпуск.";
  reference
    "RFC 8348: A YANG Data Model for Hardware Management";
}

/*
 * Идентификаторы (отождествления)
 */
identity hardware-class {
  description
    "Базовый идентификатор для всех классов оборудования.";
}

identity unknown {
  base ianahw:hardware-class;
  description
    "Неизвестный серверу класс оборудования.";
}

identity chassis {
  base ianahw:hardware-class;
  description
    "Применяется для классов, служащих контейнерами для сетевого
     оборудования. Любой класс физических компонентов,кроме стека,
     может содержаться в шасси. Шасси могут входить лишь в стек.";
}

identity backplane {
  base ianahw:hardware-class;
  description
    "Применяется для классов оборудования, представляющих
     устройства для агрегирования и пересылки сетевого трафика,
     такие как общая шина (backplane) в модульном коммутаторе
     Ethernet. Отметим, что реализация может считать шину одним
     физическим компонентом, который на деле реализован в виде
     множества дискретных физических компонентов (в шасси стека).";
}

identity container {
  base ianahw:hardware-class;
  description
    "Применяется для классов оборудования, способных включать
     съёмные физические элементы, возможно разных типов. Например,
     каждое (заполненное или пустое) гнездо шасси моделируется
     как контейнер. Отметим, что все съёмные физические компоненты
     (например, сменные модули, вентиляторы, блоки питания)
     следует моделировать внутри компонента-контейнера. Известные
```

```
        контейнеры (включая пустые) следует моделировать агенту.";
    }

    identity power-supply {
      base ianahw:hardware-class;
      description
        "Идентификатор для компонентов электропитания.";
    }

    identity fan {
      base ianahw:hardware-class;
      description
        "Идентификатор для вентиляторов и иных узлов охлаждения.";
    }

    identity sensor {
      base ianahw:hardware-class;
      description
        "Идентификатор для некоторых типов датчиков, таких как датчик
         температуры в шасси маршрутизатора.";
    }

    identity module {
      base ianahw:hardware-class;
      description
        "Идентификатор для некоторых классов самодостаточных подсистем.
         Если компонент-модуль является съёмным, его следует
         моделировать в компоненте-контейнере, иначе - напрямую в
         другом физическом компоненте (шасси или другой модуль).";
    }

    identity port {
      base ianahw:hardware-class;
      description
        "Идентификатор для сетевых портов, способных передавать или
         принимать сетевой трафик.";
    }

    identity stack {
      base ianahw:hardware-class;
      description
        "Идентификатор для суперконтейнеров (возможно, виртуальных),
         предназначенных для группировки нескольких объектов шасси.
         Стек может быть реализован виртуальным кабелем, физическим
         кабелем между разными шасси, или несколькими кабелями. Стек
         не следует моделировать внутри других физических компонентов,
         но он может входить в другой стек. В стек следует включать
         лишь шасси и другие стеки.";
    }

    identity cpu {
      base ianahw:hardware-class;
      description
        "Идентификатор для центральных процессоров.";
    }

    identity energy-object {
      base ianahw:hardware-class;
      description
        "Идентификатор оборудования, относящегося к энергии, т. е.
         элементов оборудования, являющихся частью коммуникационной
         сети (подключённого к ней), которая отслеживается,
         контролируется или предназначена для управления другим
         устройством в части потребления энергии.";
    }

    identity battery {
      base ianahw:hardware-class;
      description
        "Идентификатор батареи питания.";
    }

    identity storage-drive {
      base ianahw:hardware-class;
      description
        "Идентификатор оборудования, способного хранить данные в
         качестве основной функции, например, диск (HDD, SSD, SSHD),
         хранилище объектов (OSD) и т. п.";
    }
  }
  <CODE ENDS>
```

## 8. Взаимодействие с IANA

Этот документ задаёт исходный выпуск поддерживаемого IANA модуля YANG iana-hardware. Этот модуль предназначен для отображения модуля IANA-ENTITY-MIB, чтобы при добавлении нового перечисляемого в

IANAPhysicalClass такой же класс добавлялся как отождествление, выведенное из ianahw:hardware-class. При обновлении модуля YANG iana-hardware должен добавляться новый оператор revision перед уже имеющимися.

## 8.1. Регистрация URI

Этот документ регистрирует три URI в реестре IETF XML Registry [RFC3688] в соответствии с форматом RFC 3688.

```
URI: urn:ietf:params:xml:ns:yang:iana-hardware
Registrant Contact: The IESG.
XML: N/A, регистрируемый URI является пространством имён XML.

URI: urn:ietf:params:xml:ns:yang:ietf-hardware
Registrant Contact: The IESG.
XML: N/A, регистрируемый URI является пространством имён XML.

URI: urn:ietf:params:xml:ns:yang:ietf-hardware-state
Registrant Contact: The IESG.
XML: N/A, регистрируемый URI является пространством имён XML.
```

## 8.2. Регистрация модулей YANG

Этот документ регистрирует три модуля YANG в реестре YANG Module Names [RFC6020].

```
name:         iana-hardware
namespace:    urn:ietf:params:xml:ns:yang:iana-hardware
prefix:       ianahw
reference:    RFC 8348

name:         ietf-hardware
namespace:    urn:ietf:params:xml:ns:yang:ietf-hardware
prefix:       hw
reference:    RFC 8348

name:         ietf-hardware-state
namespace:    urn:ietf:params:xml:ns:yang:ietf-hardware-state
prefix:       hw-state
reference:    RFC 8348
```

# *9. Вопросы безопасности*

Заданные в этом документе модули YANG определяют схемы для данных, которые разработаны 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 [RFC5246].

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

В модуле YANG ietf-hardware имеется множество узлов данных, доступных для записи, создания, удаления (т. е. с принятым по умолчанию config true). Эти узлы могут быть чувствительными или уязвимыми в некоторых сетевых средах. Операции записи (например, edit-config) в такие узлы без подобающей защиты могут оказывать негативное влияние на работу сети. Ниже указаны ветви и узлы данных и их уязвимости.

***/hardware/component/admin-state***
: Установка для этого узла значения locked или shutting-down может нарушать работу служб, начиная от работающих на порту и заканчивая работой устройства в целом, в зависимости от типа компонента.

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.

***/hardware/component***
: Листья этого списка раскрывают сведения о физических компонентах устройства, которые могут служить для идентификации производителя, модели, версии и конкретных данных каждого компонента системы.

***/hardware/component/sensor-data/value***
: Эти узлы могут раскрывать значения физических датчиков устройства.

***/hardware/component/state***
: Доступ к этим узлам позволяет определить активные и ожидающие (резервные) ресурсы.

# *10. Литература*

## 10.1. 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>.

[RFC3433] Bierman, A., Romascanu, D., and K. Norseth, "Entity Sensor Management Information Base", RFC 3433, DOI 10.17487/RFC3433, December 2002, <https://www.rfc-editor.org/info/rfc3433>.

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

[RFC4268] Chisholm, S. and D. Perkins, "Entity State MIB", RFC 4268, DOI 10.17487/RFC4268, November 2005, <https://www.rfc-editor.org/info/rfc4268>.

[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>.

[RFC6933] Bierman, A., Romascanu, D., Quittek, J., and M. Chandramouli, "Entity MIB (Version 4)", RFC 6933, DOI 10.17487/RFC6933, May 2013, <https://www.rfc-editor.org/info/rfc6933>.

[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>.

[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>.

## 10.2. Дополнительная литература

[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 A. Модель данных состояния оборудования*

В этом ненормативном приложении представлено временное решение для реализаций, ещё не поддерживающих архитектуру NMDA, определённую в [RFC8342]. Ниже показана структура модуля.

```
module: ietf-hardware-state
  x--ro hardware
     x--ro last-change?   yang:date-and-time
     x--ro component* [name]
        x--ro name               string
        x--ro class              identityref
        x--ro physical-index?    int32 {entity-mib}?
        x--ro description?       string
        x--ro parent?            -> ../../component/name
        x--ro parent-rel-pos?    int32
        x--ro contains-child*    -> ../../component/name
        x--ro hardware-rev?      string
        x--ro firmware-rev?      string
        x--ro software-rev?      string
        x--ro serial-num?        string
        x--ro mfg-name?          string
        x--ro model-name?        string
        x--ro alias?             string
        x--ro asset-id?          string
        x--ro is-fru?            boolean
        x--ro mfg-date?          yang:date-and-time
        x--ro uri*               inet:uri
        x--ro uuid?              yang:uuid
        x--ro state {hardware-state}?
        |  x--ro state-last-changed?   yang:date-and-time
        |  x--ro admin-state?          hw:admin-state
        |  x--ro oper-state?           hw:oper-state
        |  x--ro usage-state?          hw:usage-state
        |  x--ro alarm-state?          hw:alarm-state
        |  x--ro standby-state?        hw:standby-state
        x--ro sensor-data {hardware-sensor}?
           x--ro value?               hw:sensor-value
           x--ro value-type?          hw:sensor-value-type
           x--ro value-scale?         hw:sensor-value-scale
           x--ro value-precision?     hw:sensor-value-precision
           x--ro oper-status?         hw:sensor-status
           x--ro units-display?       string
           x--ro value-timestamp?     yang:date-and-time
           x--ro value-update-rate?   uint32

  notifications:
    x---n hardware-state-change
    x---n hardware-state-oper-enabled {hardware-state}?
    |  x--ro name?          -> /hardware/component/name
    |  x--ro admin-state?   -> /hardware/component/state/admin-state
    |  x--ro alarm-state?   -> /hardware/component/state/alarm-state
    x---n hardware-state-oper-disabled {hardware-state}?
```

```
         x--ro name?          -> /hardware/component/name
         x--ro admin-state?   -> /hardware/component/state/admin-state
         x--ro alarm-state?   -> /hardware/component/state/alarm-state
```

## A.1. Модуль YANG для состояния оборудования

```
<CODE BEGINS> file "ietf-hardware-state@2018-03-13.yang"

module ietf-hardware-state {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-hardware-state";
  prefix hw-state;

  import ietf-inet-types {
    prefix inet;
  }
  import ietf-yang-types {
    prefix yang;
  }
  import iana-hardware {
    prefix ianahw;
  }
  import ietf-hardware {
    prefix hw;
  }

  organization
    "IETF NETMOD (Network Modeling) Working Group";

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

     Editor:   Andy Bierman
               <mailto:andy@yumaworks.com>

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

     Editor:   Jie Dong
               <mailto:jie.dong@huawei.com>

     Editor:   Dan Romascanu
               <mailto:dromasca@gmail.com>";

  description
    "Этот модуль содержит определения YANG для мониторинга
     оборудования.

     Эта модель данных разработана как временное решение для
     реализаций, ещё не поддерживающих архитектуру NMDA, заданную в
     RFC 8342. Такие реализации не могут корректно применять модуль
     ietf-hardware, поскольку без поддержки NMDA невозможно
     различать экземпляры узлов рабочей конфигурации и рабочего
     состояния.

     Модель данных в этом модуле совпадает с моделью в ietf-hardware,
     но для всех узлов задано config false.

     Если сервер, реализующий этот модуль, но не поддерживающий NMDA,
     поддерживает настройку аппаратных компонентов, ему СЛЕДУЕТ
     реализовать модуль ietf-hardware в хранилищах конфигурации.
     Соответствующие данные состояния представлены в ветви
     /hw-state:hardware.

     Авторские права (Copyright (c) 2018) 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 8348, где
     правовые вопросы рассмотрены более полно.";

  revision 2018-03-13 {
    description
      "Исходный выпуск.";
    reference
      "RFC 8348: A YANG Data Model for Hardware Management";
  }

  /*
   * Свойства (функции)
```

```
     */
    feature entity-mib {
      status deprecated;
      description
        "Указывает, что устройство реализует ENTITY-MIB.";
      reference
        "RFC 6933: Entity MIB (Version 4)";
    }

    feature hardware-state {
      status deprecated;
      description
        "Указывает, что устройство поддерживает ENTITY-STATE-MIB.";
      reference
        "RFC 4268: Entity State MIB";
    }

    feature hardware-sensor {
      status deprecated;
      description
        "Указывает, что устройство поддерживает ENTITY-SENSOR-MIB.";
      reference
        "RFC 3433: Entity Sensor Management Information Base";
    }

    /*
     * Узлы данных
     */

    container hardware {
      config false;
      status deprecated;
      description
        "Узлы данных, представляющие компоненты.";

      leaf last-change {
        type yang:date-and-time;
        status deprecated;
        description
          "Время изменения списка /hardware/component в рабочем
           состоянии.";
      }

      list component {
        key name;
        status deprecated;
        description
          "Список компонентов. Когда сервер обнаруживает новый
           аппаратный компонент, он инициализирует запись списка в
           рабочем состоянии. Если сервер не поддерживает настройку
           аппаратных компонентов, записи списка в рабочем состоянии
           инициализируются значениями для всех узлов, найденных
           реализацией. В остальных случаях выполняются указанные
           ниже процедуры.
             1. Если в списке /hardware/component предполагаемой
                (intended) конфигурации есть запись со значениями узлов
                class, parent и parent-rel-pos, равными обнаруженным,
                запись списка в рабочем состоянии, выполняются
                указанные ниже действия.
             1a. Если настроенная запись имеет значение mfg-name,
                 равное найденному, или mfg-name не удаётся найти,
                 запись в рабочем состоянии инициализируется заданными
                 значениями для всех настраиваемых узлов, включая name.
                 В иных случаях запись в списке рабочего состояния
                 инициализируется значениями всех узлов, найденными
                 реализацией. Реализация может установить сигнал,
                 указывающий, что mfg-name не соответствует условиям.
                 Способ реализации этого документ не задаёт.
             1b. В ином случае (нет соответствующей записи конфигурации)
                 записи списка в рабочем состоянии инициализируется
                 значениями для всех узлов, найденных реализацией.

           Если список /hardware/component в предполагаемой
           конфигурации изменён, система ДОЛЖНА вести себя как при
           реинициализации и следовать процедуре п. (1).";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalEntry";

        leaf name {
          type string;
          status deprecated;
          description
            "Имя компонента (может отличаться от entPhysicalName).";
        }

        leaf class {
```

```
        type identityref {
          base ianahw:hardware-class;
        }
        mandatory true;
        status deprecated;
        description
          "Оборудование общего назначения.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalClass";
      }

      leaf physical-index {
        if-feature entity-mib;
        type int32 {
          range "1..2147483647";
        }
        status deprecated;
        description
          "Представляет entPhysicalIndex для entPhysicalEntry.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalIndex";
      }

      leaf description {
        type string;
        status deprecated;
        description
          "Текстовое описание компонента. Узлу следует содержать
           строку, указывающую производителя компонента и следует
           указывать разные значения для каждой версии или модели.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalDescr";
      }

      leaf parent {
        type leafref {
          path "../../component/name";
          require-instance false;
        }
        status deprecated;
        description
          "Имя компонента, физически включающего этот компонент. Если
           этот лист не создаётся, это указывает, что компонент не
           содержится в каком-либо ином компоненте.

           Если физический компонент входит в несколько физических
           компонентов (например, модуль двойной ширины), узел
           содержит имя одного из этих компонентов. Реализация ДОЛЖНА
           использовать одно имя для каждого экземпляра узла.";
        reference
          "RFC 6933: Entity MIB (Version 4) -
                     entPhysicalContainedIn";
      }

      leaf parent-rel-pos {
        type int32 {
          range "0 .. 2147483647";
        }
        status deprecated;
        description
          "Указывает относительное положение дочернего компонента
           среди братских, к каковым относятся компоненты, имеющие
           одно значение узла parent и одно базовое отождествление
           для узла class. Отметим, что второе правило обеспечивает
           реализациям гибкость нумерации компонентов. Например,
           некоторые реализации могут иметь одну серию номеров для
           всех компонентов, производных от ianahw:port, а другие -
           применять для них разные серии номеров (скажем, одну для
           разъёмов RJ45, другую для SFP).";

        reference
          "RFC 6933: Entity MIB (Version 4) -
                     entPhysicalParentRelPos";
      }

      leaf-list contains-child {
        type leafref {
          path "../../component/name";
        }
        status deprecated;
        description
          "Имя содержащегося компонента.";
        reference
          "RFC 6933: Entity MIB (Version 4) - entPhysicalChildIndex";
      }
```

```
        leaf hardware-rev {
          type string;
          status deprecated;
          description
            "Фирменная строка аппаратного выпуска компонента.
             Предпочтительно указывать идентификатор, помещаемый
             производителем на сам компонент (при наличии).";
          reference
            "RFC 6933: Entity MIB (Version 4) -
                       entPhysicalHardwareRev";
        }

        leaf firmware-rev {
          type string;
          status deprecated;
          description
            "Фирменная строка выпуска микрокода для компонента.";
          reference
            "RFC 6933: Entity MIB (Version 4) -
                       entPhysicalFirmwareRev";
        }

        leaf software-rev {
          type string;
          status deprecated;
          description
           "Фирменная строка программного выпуска компонента.";
          reference
            "RFC 6933: Entity MIB (Version 4) -
                       entPhysicalSoftwareRev";
        }

        leaf serial-num {
          type string;
          status deprecated;
          description
            "Фирменная строка серийного номера компонента.
             Предпочтительно указывать номер, помещаемый
             производителем на сам компонент (при наличии).";
          reference
            "RFC 6933: Entity MIB (Version 4) - entPhysicalSerialNum";
        }

        leaf mfg-name {
          type string;
          status deprecated;
          description
            "Название изготовителя для физического компонента.
             Предпочтительно указывать название, помещаемое
             изготовителем на сам компонент (при наличии).";

             Отметим, что сравнение значений model-name, firmware-rev,
             software-rev и serial-num имеет смысл лишь для
             компонентов с одним значением mfg-name.

             Если заданное изготовителем название физического
             компонента неизвестно серверу, этот узел не создаётся.";
          reference
            "RFC 6933: Entity MIB (Version 4) - entPhysicalMfgName";
        }

        leaf model-name {
          type string;
          status deprecated;
          description
            "Фирменная строка идентификатора модели физического
             компонента. Предпочтительно указывать видимый клиенту
             код, напечатанный на самом компоненте (при наличии).

             Если заданный производителем код модели физического
             компонента неизвестно серверу, этот узел не создаётся.";
          reference
            "RFC 6933: Entity MIB (Version 4) - entPhysicalModelName";
        }

        leaf alias {
          type string;
          status deprecated;
          description
            "Псевдоним имени компонента, заданный администратором сети,
             сохраняющийся при выключении или перезагрузке компонента.

             Если псевдоним не задан, сервер МОЖЕТ установить для узла
             локально уникальное значение в рабочем состоянии.

             Реализация сервера МОЖЕТ сопоставить этот лист с объектом
```

```
        MIB entPhysicalAlias. Такой реализации нужен механизм для
        обработки различий в размере и разрешённых символах между
        этим листом и entPhysicalAlias. Задание такого механизма
        выходит за рамки этого документа.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalAlias";
  }

  leaf asset-id {
    type string;
    status deprecated;
    description
      "Заданный пользователем идентификатор отслеживания активов
       для компонента.

       Реализация сервера МОЖЕТ сопоставить этот лист с объектом
       MIB entPhysicalAssetID. Такой реализации нужен механизм
       обработки различий в размере и разрешённых символах между
       этим листом и entPhysicalAssetID. Задание такого механизма
       выходит за рамки этого документа.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalAssetID";
  }

  leaf is-fru {
    type boolean;
    status deprecated;
    description
      "Указывает, считает ли производитель этот компонент
       заменяемым в полевых условиях. Значение true указывает
       возможность такой замены. Для всех компонентов, постоянно
       содержащихся в этом компоненте для этого узла следует
       возвращать значение false.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalIsFRU";
  }

  leaf mfg-date {
    type yang:date-and-time;
    status deprecated;
    description
      "Дата изготовления компонента.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalMfgDate";
  }

  leaf-list uri {
    type inet:uri;
    status deprecated;
    description
      "Идентификационные сведения о компоненте.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalUris";
  }

  leaf uuid {
    type yang:uuid;
    status deprecated;
    description
      "Глобально уникальный идентификатор (UUID) компонента.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entPhysicalUUID";
  }

  container state {
    if-feature hardware-state;
    status deprecated;
    description
      "Относящиеся к состоянию узлы.";
    reference
      "RFC 4268: Entity State MIB";

    leaf state-last-changed {
      type yang:date-and-time;
      status deprecated;
      description
        "Дата и время смены любого из значений admin-state,
         oper-state, usage-state, alarm-state или
         standby-state для этого компонента.

         Если с момента последней реинициализации системы не было
         изменений, узел содержит дату и время инициализации
         локальной системы. Если не было изменений с момента
         установки компонента в локальную систему, указывается
         дата и время такой установки.";
      reference
```

```
          "RFC 4268: Entity State MIB - entStateLastChanged";
      }

      leaf admin-state {
        type hw:admin-state;
        status deprecated;
        description
          "Административное состояние компонента, управляющее
           обслуживанием другие компонентов в иерархии
           содержания, а также других пользователей этих услуг,
           заданных мерами, выходящими за рамки этого модуля.

           Некоторые компоненты раскрывают лишь часть остальных
           значений административного состояния. Некоторые
           компоненты нельзя блокировать, поэтому узел показывает
           лишь статус unlocked. Другие компоненты нельзя аккуратно
           отключить и узел не показывает статус shutting-down.";
        reference
          "RFC 4268: Entity State MIB - entStateAdmin";
      }

      leaf oper-state {
        type hw:oper-state;
        status deprecated;
        description
          "Рабочее состояние компонента. Отметим, что этот узел не
           за административным статусом, т. е. административный
           статус down не означает рабочего статуса disabled.

           Отметим, что некоторые реализации могут не поддерживать
           точное отображение oper-state, когда admin-state для
           узла отличается от unlocked. В таких случаях узел ДОЛЖЕН
           иметь значение unknown.";
        reference
          "RFC 4268: Entity State MIB - entStateOper";
      }

      leaf usage-state {
        type hw:usage-state;
        status deprecated;
        description
          "Состояние использования компонента, отражающее его
           способность обслуживать дополнительные компоненты в
           иерархии содержания.

           Некоторые компоненты раскрывают лишь часть значений
           статуса использования. Компоненты, не способные
           обслуживать какие-либо компоненты в иерархии размещения,
           всегда будут иметь статус busy. В некоторых случаях
           компонент может поддерживать лишь ещё один компонент в
           иерархии размещения и будет раскрывать лишь значение
           idle или busy.";
        reference
          "RFC 4268: Entity State MIB - entStateUsage";
      }

      leaf alarm-state {
        type hw:alarm-state;
        status deprecated;
        description
          "Состояние сигналов тревоги для компонента без учёта
           сигналов в дочерних компонентах иерархии размещения.";
        reference
          "RFC 4268: Entity State MIB - entStateAlarm";
      }

      leaf standby-state {
        type hw:standby-state;
        status deprecated;
        description
          "Состояние ожидания (standby) для компонента.

           Некоторые компоненты раскрывают лишь часть значений
           статуса ожидания. Компоненты, не способные работать
           в режиме ожидания будут иметь для этого узла значение
           providing-service.";
        reference
          "RFC 4268: Entity State MIB - entStateStandby";
      }
    }

    container sensor-data {
      when 'derived-from-or-self(../class,
                                 "ianahw:sensor")' {
        description
          "Узлы данных для компонентов типа sensor";
```

```
        }
        if-feature hardware-sensor;
        status deprecated;

        description
          "Связанные с датчиками узлы.";
        reference
          "RFC 3433: Entity Sensor Management Information Base";

        leaf value {
          type hw:sensor-value;
          status deprecated;
          description
            "Последнее измерение, полученное сервером от датчика.

             Клиенту, периодически просматривающему этот узел,
             следует извлекать и узлы value-type, value-scale и
             value-precision, поскольку они тоже могут изменяться.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorValue";
        }

        leaf value-type {
          type hw:sensor-value-type;
          status deprecated;
          description
            "Тип единиц, связанных со значением датчика";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorType";
        }

        leaf value-scale {
          type hw:sensor-value-scale;
          status deprecated;
          description
            "Коэффициент (степень 10) измерения для значения узла";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorScale";
        }

        leaf value-precision {
          type hw:sensor-value-precision;
          status deprecated;
          description
            "Число десятичных знаков точности значения узла.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorPrecision";
        }

        leaf oper-status {
          type hw:sensor-status;
          status deprecated;
          description
            "Рабочее состояние датчика.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorOperStatus";
        }

        leaf units-display {
          type string;
          status deprecated;
          description
            "Текстовое описание единиц, которые следует применять при
             выводе значения датчика.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorUnitsDisplay";
        }

        leaf value-timestamp {
          type yang:date-and-time;
          status deprecated;
          description
            "Время последнего извлечения сервером статуса или
             значения этого датчика.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorValueTimeStamp";
        }

        leaf value-update-rate {
```

```
          type uint32;
          units "milliseconds";
          status deprecated;
          description
            "Интервал, с которым сервер обновляет связанный узел
             value в миллисекундах. Значение 0 указывает, что:

              - значение датчика обновляется по запросу (например,
                при опросе сервером для get-request);

              - значение датчика обновляется при его изменении
                (по событию);

              - сервер не знает частоты обновления.";
          reference
            "RFC 3433: Entity Sensor Management Information Base -
                       entPhySensorValueUpdateRate";
        }
      }
    }
  }

  /*
   * Уведомления
   */

  notification hardware-state-change {
    status deprecated;
    description
     "Уведомление hardware-state-change создается при изменении
      значения /hardware/last-change в рабочем состоянии.";
    reference
      "RFC 6933: Entity MIB (Version 4) - entConfigChange";
  }

  notification hardware-state-oper-enabled {
    if-feature hardware-state;
    status deprecated;
    description
      "Уведомление hardware-state-oper-enabled указывает переход
       компонента в состояние enabled.";

    leaf name {
      type leafref {
        path "/hardware/component/name";
      }
      status deprecated;
      description
        "Имя компонента, перешедшего в состояние enabled.";
    }
    leaf admin-state {
      type leafref {
        path "/hardware/component/state/admin-state";
      }
      status deprecated;
      description
        "Административное состояние компонента.";
    }
    leaf alarm-state {
      type leafref {
        path "/hardware/component/state/alarm-state";
      }
      status deprecated;
      description
        "Состояние сигналов тревоги для компонента.";
    }
    reference
      "RFC 4268: Entity State MIB - entStateOperEnabled";
  }

  notification hardware-state-oper-disabled {
    if-feature hardware-state;
    status deprecated;
    description
      "Уведомление hardware-state-oper-disabled указывает переход
       компонента в состояние disabled.";

    leaf name {
      type leafref {
        path "/hardware/component/name";
      }
      status deprecated;
      description
        "Имя компонента, перешедшего в состояние disabled.";
    }
    leaf admin-state {
```

```
        type leafref {
          path "/hardware/component/state/admin-state";
        }
        status deprecated;
        description
          "Административное состояние компонента.";
      }
      leaf alarm-state {
        type leafref {
          path "/hardware/component/state/alarm-state";
        }
        status deprecated;
        description
          "Состояние сигналов тревоги для компонента.";
      }
      reference
        "RFC 4268: Entity State MIB - entStateOperDisabled";
    }

  }
  <CODE ENDS>
```

## Благодарности

Авторы благодарны Bart Bogaert, Timothy Carey, William Lupton и Juergen Schoenwaelder за комментарии к предварительным вариантам этого документа.

## Адреса авторов

**Andy Bierman**
YumaWorks
Email: andy@yumaworks.com

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

**Jie Dong**
Huawei Technologies
Email: jie.dong@huawei.com

**Dan Romascanu**
Email: dromasca@gmail.com

**Перевод на русский язык**

Николай Малых

nmalykh@protokols.ru