OPSAWG
Internet-Draft
Intended status: Standards Track
Expires: October 25, 2021

B. Claise
Huawei
J. Quilbeuf
Independent
P. Lucente
NTT
P. Fasano
TIM S.p.A
T. Arumugam
Cisco Systems, Inc.

April 23, 2021

YANG Modules for Service Assurance
draft-claise-opsawg-service-assurance-yang-07

Abstract

This document proposes YANG modules for the Service Assurance for Intent-based Networking Architecture.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 25, 2021.

Table of Contents

1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The terms used in this document are defined in draft-claise-opsawg-service-assurance-architecture IETF draft.

2. Introduction

The "Service Assurance for Intent-based Networking Architecture" draft-claise-opsawg-service-assurance-architecture, specifies the framework and all of its components for service assurance. This document complements the architecture by providing open interfaces between components. More specifically, the goal is to provide YANG modules for the purpose of service assurance in a format that is:

- machine readable
- vendor independent
- augmentable

3. YANG Models Overview

The main YANG module, ietf-service-assurance, defines objects for assuring network services based on their decomposition into so-called subservices. The subservices are hierarchically organised by dependencies. The subservices, along with the dependencies, constitute an assurance graph. This module should be supported by an agent, able to interact with the devices in order to produce a health status and symptoms for each subservice in the assurance graph. This module is intended for the following use cases:

- Assurance graph configuration:
  - Subservices: configure a set of subservices to assure, by specifying their types and parameters.
  - Dependencies: configure the dependencies between the subservices, along with their type.
- Assurance telemetry: export the health status of the subservices, along with the observed symptoms.

The second YANG module, ietf-service-assurance-device, extends the ietf-service-assurance module to add support for the subservice DeviceHealthy. Additional subservice types might be added the same way.

The third YANG module, example-service-assurance-device-acme, extends the ietf-service-assurance-device module as an example to add support for the subservice DeviceHealthy, with specifics for the fictional ACME Corporation. Additional vendor-specific parameters might be added the same way.

4. Base ietf-service-assurance YANG module

4.1. Tree View

The following tree diagram [RFC8340] provides an overview of the ietf-service-assurance data model.

```
module: ietf-service-assurance
  +--ro assurance-graph-version        yang:counter32
  +--ro assurance-graph-last-change    yang:date-and-time
  +--rw subservices
     +--rw subservice* [type id]
        +--rw type                                identityref
        +--rw id                                  string
        +--ro last-change?                        yang:date-and-time
        +--ro label?                              string
        +--rw under-maintenance?                  boolean
        +--rw maintenance-contact                 string
        +--rw (parameter)?
        |  +--:(service-instance-parameter)
        |     +--rw service-instance-parameter
        |        +--rw service?         string
        |        +--rw instance-name?   string
        +--ro health-score?                       uint8
        +--ro symptoms-history-start?             yang:date-and-time
        +--rw symptoms
        |  +--ro symptom* [start-date-time id]
        |     +--ro id                      string
        |     +--ro health-score-weight?    uint8
        |     +--ro description?            string
        |     +--ro start-date-time         yang:date-and-time
        |     +--ro stop-date-time?         yang:date-and-time
        +--rw dependencies
           +--rw dependency* [type id]
              +--rw type               -> /subservices/subservice/type
              +--rw id                 -> /subservices/
subservice[type=current()/../type]/id
```

```
            +--rw dependency-type?   identityref
```

4.2. Concepts

The ietf-service-assurance YANG model assumes an identified number of subservices, to be assured independently. A subservice is a feature or a subpart of the network system that a given service instance might depend on. Example of subservices include:

- DeviceHealthy: whether a device is healthy, and if not, what are the symptoms. Potential symptoms are "CPU overloaded", "Out of RAM", or "Out of TCAM".

- ConnectivityHealthy: given two IP addresses owned by two devices, what is the quality of the connection between them. Potential symptoms are "No route available" or "ECMP Imbalance".

The first example is a subservice representing a subpart of the network system, while the second is a subservice representing a feature of the network, In both cases, these subservices might depend on other subservices, for instance, the connectivity might depend on a subservice representing the routing mechanism and on a subservice representing ECMP.

The symptoms are listed for each subservice. Each symptom is specified by a unique id and contains a health-score-weight (the impact to the health score incurred by this symptom), a label (text describing what the symptom is), and dates and times at which the symptom was detected and stopped being detected. While the unique id is sufficient as an unique key list, the start-date-time second key help sorting and retrieving relevant symptoms.

The assurance of a given service instance can be obtained by composing the assurance of the subservices that it depends on, via the dependency relations.

In order to declare a subservice MUST provide:

- A type: identity inheriting of the base identity for subservice,

- An id: string uniquely identifying the subservice among those with the same identity,

- Some parameters, which should be specified in an augmenting model, as described in the next sections.

The type and id uniquely identify a given subservice. They are used to indicate the dependencies. Dependencies have types as well. Two types are specified in the model:

- Impacting: such a dependency indicates an impact on the health of the dependent,

- Informational: such a dependency might explain why the dependent has issues but does not impact its health.

To illustrate the difference between "impacting" and "informational", consider the subservice InterfaceHealthy, representing a network interface. If the device to which the network interface belongs goes down, the network interface will transition to a down state as well. Therefore, the dependency of InterfaceHealthy towards DeviceHealthy is "impacting". On the other hand, as a the dependency towards the ECMPLoad subservice, which checks that the load between ECMP remains ce remains stable throughout time, is only "informational". Indeed, services might be perfectly healthy even if the load distribution between ECMP changed. However, such an instability might be a relevant symptom for diagnosing the root cause of a problem.

Service instances MUST be modeled as a particular type of subservice with two parameters, a type and an instance name. The type is the name of the service defined in the network orchestrator, for instance "point-to-point-l2vpn". The instance name is the name assigned to the particular instance that we are assuring, for instance the name of the customer using that instance.

The "under-maintenance" and "maintenance-contact" flags inhibit the emission of symptoms for that subservice and subservices that depend on them. See Section 3.7 of [draft-claise-opsawg-service-assurance-architecture] for a more detailed discussion.

By specifying service instances and their dependencies in terms of subservices, one defines the whole assurance to apply for them. An assurance agent supporting this model should then produce telemetry in return with, for each subservice: a health-status indicating how healthy the subservice is and when the subservice is not healthy, a list of symptoms explaining why the subservice is not healthy.

4.3. YANG Module

<CODE BEGINS> file "ietf-service-assurance@2020-01-13.yang"

```
module ietf-service-assurance {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-service-assurance";
  prefix service-assurance;

  import ietf-yang-types {
```

```
    prefix yang;
  }

  organization
    "IETF NETCONF (Network Configuration) Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/netconf/>
     WG List:  <mailto:netconf@ietf.org>
     Author:   Benoit Claise  <mailto:bclaise@cisco.com>
     Author:   Jean Quilbeuf   <mailto:jquilbeu@cisco.com>";
  description
    "This module defines objects for assuring network services based on
    their decomposition into so-called subservices, according to the SAIN
    (Service Assurance for Intent-based Networking) architecture.

    The subservices hierarchically organised by dependencies constitute an
    assurance graph. This module should be supported by an assurance agent,
    able to interact with the devices in order to produce a health status
    and symptoms for each subservice in the assurance graph.

    This module is intended for the following use cases:
    * Assurance graph configuration:
      * subservices: configure a set of subservices to assure, by specifying
        their types and parameters.
      * dependencies: configure the dependencies between the subservices,
         along with their type.
    * Assurance telemetry: export the health status of the subservices, along
      with the observed symptoms.

    The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
    'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
    'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document
    are to be interpreted as described in BCP 14 (RFC 2119)
    (RFC 8174) when, and only when, they appear in all
    capitals, as shown here.

    Copyright (c)2020 IETF Trust and the persons identified as
    authors of the code.  All rights reserved.

    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD License
    set forth in Section 4.c of the IETF Trust's Legal Provisions
    Relating to IETF Documents
    (https://trustee.ietf.org/license-info).

    This version of this YANG module is part of RFC XXXX; see the
    RFC itself for full legal notices.
```

```
    TO DO:
    - Better type (IETF or OC) for device-id, interface-id, etc.
    - Have a YANG module for IETF and one for OC?";

  revision 2020-01-13 {
    description
      "Added the maintenance window concept.";
    reference
      "RFC xxxx: Title to be completed";
  }

  revision 2019-11-16 {
    description
      "Initial revision.";
    reference
      "RFC xxxx: Title to be completed";
  }

  identity subservice-idty {
    description
      "Root identity for all subservice types.";
  }

  identity service-instance-idty {
    base subservice-idty;
    description
      "Identity representing a service instance.";
  }

  identity dependency-type {
    description
      "Base identity for representing dependency types.";
  }

  identity informational-dependency {
    base dependency-type;
    description
      "Indicates that symptoms of the dependency might be of interest for the
       dependent, but the status of the dependency should not have any
       impact on the dependent.";
  }

  identity impacting-dependency {
    base dependency-type;
    description
      "Indicates that the status of the dependency directly impacts the status
       of the dependent.";
  }
```

```
  grouping symptom {
    description
      "Contains the list of symptoms for a specific subservice.";
    leaf id {
      type string;
      description
        "A unique identifier for the symptom.";
    }
    leaf health-score-weight {
      type uint8 {
        range "0 .. 100";
      }
      description
        "The weight to the health score incurred by this symptom. The higher
the
         value, the more of an impact this symptom has. If a subservice health
         score is not 100, there must be at least one symptom with a health
         score weight larger than 0.";
    }
    leaf description {
      type string;
      description
        "Description of the symptom, i.e. text describing what the symptom is,
to
         be computer-consumable and be displayed on a human interface. ";
    }
    leaf start-date-time {
      type yang:date-and-time;
      description
        "Date and time at which the symptom was detected.";
    }
    leaf stop-date-time {
      type yang:date-and-time;
      description
        "Date and time at which the symptom stopped being detected.";
    }
  }

  grouping subservice-dependency {
    description
      "Represent a dependency to another subservice.";
    leaf type {
      type leafref {
        path "/subservices/subservice/type";
      }
      description
        "The type of the subservice to refer to (e.g. DeviceHealthy).";
    }
```

```
    leaf id {
      type leafref {
```

```
        path "/subservices/subservice[type=current()/../type]/id";
      }
      description
        "The identifier of the subservice to refer to.";
    }
    leaf dependency-type {
      type identityref {
        base dependency-type;
      }
      description
        "Represents the type of dependency (i.e. informational, impacting).";
    }
    // augment here if more info are needed (i.e. a percentage) depending on
the dependency type.
  }

  leaf assurance-graph-version {
    type yang:counter32;
    mandatory true;
    config false;
    description
      "The assurance graph version, which increases by 1 for each new version,
after the changes
       (dependencies and/or maintenance windows parameters) are applied to the
subservice(s).";
  }
  leaf assurance-graph-last-change {
    type yang:date-and-time;
    mandatory true;
    config false;
    description
      "Date and time at which the assurance graph last changed after the
changes (dependencies
       and/or maintenance windows parameters) are applied to the subservice(s).
These date and time
       must be more recent or equal compared to the more recent value of any
changed subservices
       last-change";
  }
  container subservices {
    description
      "Root container for the subservices.";
    list subservice {
      key "type id";
      description
        "List of subservice configured.";
      leaf type {
        type identityref {
```

```
          base subservice-idty;
        }
        description
          "Name of the subservice, e.g. DeviceHealthy.";
      }
      leaf id {
```

```
        type string;
        description
          "Unique identifier of the subservice instance, for each type.";
      }
      leaf last-change {
        type yang:date-and-time;
        config false;
        description
          "Date and time at which the assurance graph for this subservice
           instance last changed, i.e. dependencies and/or maintenance windows
parameters.";
      }
      leaf label {
        type string;
        config false;
        description
          "Label of the subservice, i.e. text describing what the subservice is
to
           be displayed on a human interface.";
      }
      leaf under-maintenance {
        type boolean;
        default false;
        description
          "An optional flag indicating whether this particular subservice is
under
          maintenance. Under this circumstance, the subservice symptoms and the
          symptoms of its dependencies in the assurance graph should not be
taken
          into account. Instead, the subservice should send a 'Under
Maintenance'
          single symptom.

          The operator changing the under-maintenance value must set the
          maintenance-contact variable.

          When the subservice is not under maintenance any longer, the
          under-maintenance flag must return to its default value and
          the under-maintenance-owner variable deleted.";
      }
      leaf maintenance-contact {
        when "../under-maintenance = 'true'";
        type string;
        mandatory true;
        description
          "A string used to model an administratively assigned name of the
          resource that changed the under-maintenance value to 'true.
```

```
          It is suggested that this name contain one or more of the following:
          IP address, management station name, network manager's name,
location,
          or phone number. In some cases the agent itself will be the owner of
          an entry. In these cases, this string shall be set to a string
          starting with 'monitor'.";
```

```
      }
      choice parameter {
        description
          "Specify the required parameters per subservice type.";
        container service-instance-parameter {
          when "derived-from-or-self(../type, 'service-assurance:service-
instance-idty')";
          description
            "Specify the parameters of a service instance.";
          leaf service {
            type string;
            description "Name of the service.";
          }
          leaf instance-name{
            type string;
            description "Name of the instance for that service.";
          }
        }
        // Other modules can augment their own cases into here
      }
      leaf health-score {
        type uint8 {
          range "0 .. 100";
        }
        config false;
        description
           "Score value of the subservice health. A value of 100 means that
           subservice is healthy. A value of 0 means that the subservice is
           broken. A value between 0 and 100 means that the subservice is
           degraded.";
      }
       leaf symptoms-history-start {
        type yang:date-and-time;
        config false;
        description
           "Date and time at which the symptoms history starts for this
           subservice instance, either because the subservice instance
           started at that date and time or because the symptoms before that
           were removed due to a garbage collection process.";
      }
      container symptoms {
        description
          "Symptoms for the subservice.";
        list symptom {
          key "start-date-time id";
          config false;
          description
            "List of symptoms the subservice. While the start-date-time key is
```

not
            necessary per se, this would get the entries sorted by start-date-
time

```
            for easy consumption.";
          uses symptom;
        }
      }
      container dependencies {
        description
          "configure the dependencies between the subservices, along with their
types.";
        list dependency {
          key "type id";
          description
            "List of soft dependencies of the subservice.";
          uses subservice-dependency;
        }
      }
    }
  }
}
```

<CODE ENDS>

5. Subservice Extension: ietf-service-assurance-device YANG module

5.1. Tree View

The following tree diagram [RFC8340] provides an overview of the ietf-service-assurance-device data model.

```
module: ietf-service-assurance-device
  augment /service-assurance:subservices/service-assurance:subservice/service-
assurance:parameter:
    +--rw device-idty
       +--rw device?   string
```

5.2. Complete Tree View

The following tree diagram [RFC8340] provides an overview of the ietf-service-assurance and ietf-service-assurance-device data models.

```
module: ietf-service-assurance
  +--ro assurance-graph-version        yang:counter32
  +--ro assurance-graph-last-change    yang:date-and-time
  +--rw subservices
     +--rw subservice* [type id]
        +--rw type                                          identityref
        +--rw id                                            string
        +--ro last-change?                                  yang:date-and-time
        +--ro label?                                        string
        +--rw under-maintenance?                            boolean
        +--rw maintenance-contact                           string
        +--rw (parameter)?
        |  +--:(service-instance-parameter)
        |  |  +--rw service-instance-parameter
        |  |     +--rw service?         string
        |  |     +--rw instance-name?   string
        |  +--:(service-assurance-device:device-idty)
        |     +--rw service-assurance-device:device-idty
        |        +--rw service-assurance-device:device?   string
        +--ro health-score?                                 uint8
        +--ro symptoms-history-start?                       yang:date-and-time
        +--rw symptoms
        |  +--ro symptom* [start-date-time id]
        |     +--ro id                     string
        |     +--ro health-score-weight?   uint8
        |     +--ro description?           string
        |     +--ro start-date-time        yang:date-and-time
        |     +--ro stop-date-time?        yang:date-and-time
        +--rw dependencies
           +--rw dependency* [type id]
              +--rw type               -> /subservices/subservice/type
              +--rw id                 -> /subservices/
subservice[type=current()/../type]/id
              +--rw dependency-type?   identityref
```

5.3. Concepts

As the number of subservices will grow over time, the YANG module is designed to be extensible. A new subservice type requires the precise specifications of its type and expected parameters. Let us illustrate the example of the new DeviceHealthy subservice type. As the name implies, it monitors and reports the device health, along with some symptoms in case of degradation.

For our DeviceHealthy subservice definition, the new device-idty is specified, as an inheritance from the base identity for subservices. This indicates to the assurance agent that we are now assuring the health of a device.

The typical parameter for the configuration of the DeviceHealthy subservice is the name of the device that we want to assure. By augmenting the parameter choice from ietf-service-assurance YANG module for the case of the device-idty subservice type, this new parameter is specified.

5.4. YANG Module

<CODE BEGINS> file "ietf-service-assurance-device@2020-01-13.yang"

```
module ietf-service-assurance-device {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-service-assurance-device";
  prefix service-assurance-device;

  import ietf-service-assurance {
    prefix "service-assurance";
  }

  organization
    "IETF NETCONF (Network Configuration) Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/netconf/>
     WG List:  <mailto:netconf@ietf.org>
     Author:   Benoit Claise  <mailto:bclaise@cisco.com>
     Author:   Jean Quilbeuf   <mailto:jquilbeu@cisco.com>";
  description
   "This module extends the ietf-service-assurance module to add
    support for the subservice DeviceHealthy.

    Checks whether a network device is healthy.

    The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
    'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
    'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document
    are to be interpreted as described in BCP 14 (RFC 2119)
    (RFC 8174) when, and only when, they appear in all
    capitals, as shown here.

    Copyright (c) 2020 IETF Trust and the persons identified as
    authors of the code.  All rights reserved.

    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD License
    set forth in Section 4.c of the IETF Trust's Legal Provisions
    Relating to IETF Documents
    (https://trustee.ietf.org/license-info).
```

```
    This version of this YANG module is part of RFC XXXX; see the
    RFC itself for full legal notices.  ";

  revision 2020-01-13 {
    description
      "Added the maintenance window concept.";
    reference
      "RFC xxxx: Title to be completed";
  }

  revision 2019-11-16 {
    description
      "Initial revision.";
    reference
        "RFC xxxx: Title to be completed";
  }

  identity device-idty {
    base service-assurance:subservice-idty;
    description "Network Device is healthy.";
  }

  augment /service-assurance:subservices/service-assurance:subservice/service-
assurance:parameter {
       description
         "Specify the required parameters for a new subservice type";
       container device-idty{
           when "derived-from-or-self(../service-assurance:type, 'device-
idty')";
           description
             "Specify the required parameters for the device-idty subservice
type";

           leaf device {
               type string;
               description "The device to monitor.";
           }
       }
  }
}
```

<CODE ENDS>

6. Subservice Extension: ietf-service-assurance-interface YANG module

6.1. Tree View

The following tree diagram [RFC8340] provides an overview of the ietf-service-assurance-interface data model.

```
module: ietf-service-assurance-interface
  augment /service-assurance:subservices/service-assurance:subservice/service-
assurance:parameter:
    +--rw device?      string
    +--rw interface?   string
```

6.2. Complete Tree View

The following tree diagram [RFC8340] provides an overview of the ietf-service-assurance, ietf-service-assurance-device, and ietf-service-assurance-interface data models.

```
module: ietf-service-assurance
  +--ro assurance-graph-version        yang:counter32
  +--ro assurance-graph-last-change    yang:date-and-time
  +--rw subservices
     +--rw subservice* [type id]
        +--rw type                                          identityref
        +--rw id                                            string
        +--ro last-change?                                  yang:date-and-time
        +--ro label?                                        string
        +--rw under-maintenance?                            boolean
        +--rw maintenance-contact                           string
        +--rw (parameter)?
        |  +--:(service-instance-parameter)
        |  |  +--rw service-instance-parameter
        |  |     +--rw service?         string
        |  |     +--rw instance-name?   string
        |  +--:(service-assurance-device:device-idty)
        |  |  +--rw service-assurance-device:device-idty
        |  |     +--rw service-assurance-device:device?   string
        |  +--:(service-assurance-interface:device)
        |  |  +--rw service-assurance-interface:device?      string
        |  +--:(service-assurance-interface:interface)
        |     +--rw service-assurance-interface:interface?   string
        +--ro health-score?                                 uint8
        +--ro symptoms-history-start?                       yang:date-and-time
        +--rw symptoms
        |  +--ro symptom* [start-date-time id]
        |     +--ro id                     string
        |     +--ro health-score-weight?   uint8
        |     +--ro description?           string
        |     +--ro start-date-time        yang:date-and-time
        |     +--ro stop-date-time?        yang:date-and-time
        +--rw dependencies
           +--rw dependency* [type id]
              +--rw type              -> /subservices/subservice/type
              +--rw id                -> /subservices/
```

```
subservice[type=current()/../type]/id
              +--rw dependency-type?   identityref
```

6.3. Concepts

For our InterafaceHealthy subservice definition, the new interface-idty is specified, as an inheritance from the base identity for subservices. This indicates to the assurance agent that we are now assuring the health of an interface.

The typical parameters for the configuration of the InterfaceHealthy subservice are the name of the device and, on that specific device, a specific interface. By augmenting the parameter choice from ietf-service-assurance YANG module for the case of the interface-idty subservice type, those two new parameter are specified.

6.4. YANG Module

<CODE BEGINS> file "ietf-service-assurance-interface@2020-01-13.yang"

```
module ietf-service-assurance-interface {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-service-assurance-interface";
  prefix service-assurance-interface;

  import ietf-service-assurance {
    prefix "service-assurance";
  }

  organization
    "IETF OPSAWG Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
     WG List:  <mailto:opsawg@ietf.org>
     Author:   Benoit Claise  <mailto:bclaise@cisco.com>
     Author:   Jean Quilbeuf   <mailto:jquilbeu@cisco.com>";
  description
   "This module extends the ietf-service-assurance module to add
    support for the subservice InterfaceHealthy.

    Checks whether an interface is healthy.

    The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
    'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
    'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document
    are to be interpreted as described in BCP 14 (RFC 2119)
    (RFC 8174) when, and only when, they appear in all
    capitals, as shown here.

    Copyright (c) 2020 IETF Trust and the persons identified as
    authors of the code.  All rights reserved.
```

This version of this YANG module is part of RFC XXXX; see the
RFC itself for full legal notices. ";

```
  revision 2020-01-13 {
    description
      "Initial revision.";
    reference
        "RFC xxxx: Title to be completed";
  }

  identity interface-idty {
    base service-assurance:subservice-idty;
    description "Checks whether an interface is healthy.";
  }

  augment /service-assurance:subservices/service-assurance:subservice/service-
assurance:parameter {
       when "derived-from-or-self(service-assurance:type, 'interface-idty')";
       description
         "Specify the required parameters for the interface-idty subservice
type";

       leaf device {
           type string;
           description "Device supporting the interface.";
       }
       leaf interface {
           type string;
           description "Name of the interface.";
       }
  }
}
```

<CODE ENDS>

7. Vendor-specific Subservice Extension: example-service-assurance-device-acme YANG module

7.1. Tree View

The following tree diagram [RFC8340] provides an overview of the example-service-assurance-device-acme data model.

```
module: example-service-assurance-device-acme
  augment /service-assurance:subservices/service-assurance:subservice/service-
assurance:parameter:
    +--rw acme-device-idty
       +--rw device?                    string
       +--rw acme-specific-parameter?   string
```

7.2. Complete Tree View

The following tree diagram [RFC8340] provides an overview of the ietf-service-assurance, ietf-service-assurance-device, and example-service-assurance-device-acme data models.

```
module: ietf-service-assurance
  +--ro assurance-graph-version        yang:counter32
  +--ro assurance-graph-last-change    yang:date-and-time
  +--rw subservices
     +--rw subservice* [type id]
        +--rw type
identityref
        +--rw id
string
        +--ro last-change?
yang:date-and-time
        +--ro label?
string
        +--rw under-maintenance?
boolean
        +--rw maintenance-contact
string
        +--rw (parameter)?
        |  +--:(service-instance-parameter)
        |  |  +--rw service-instance-parameter
        |  |     +--rw service?         string
        |  |     +--rw instance-name?   string
        |  +--:(service-assurance-device:device-idty)
        |  |  +--rw service-assurance-device:device-idty
        |  |     +--rw service-assurance-device:device?   string
        |  +--:(service-assurance-interface:device)
        |  |  +--rw service-assurance-interface:device?
string
        |  +--:(service-assurance-interface:interface)
        |  |  +--rw service-assurance-interface:interface?
string
        |  +--:(example-service-assurance-device-acme:acme-device-idty)
        |     +--rw example-service-assurance-device-acme:acme-device-idty
        |        +--rw example-service-assurance-device-
acme:device?                    string
        |        +--rw example-service-assurance-device-acme:acme-specific-
parameter?   string
        +--ro health-score?
uint8
        +--ro symptoms-history-start?
yang:date-and-time
        +--rw symptoms
        |  +--ro symptom* [start-date-time id]
        |     +--ro id                     string
        |     +--ro health-score-weight?   uint8
        |     +--ro description?           string
        |     +--ro start-date-time        yang:date-and-time
        |     +--ro stop-date-time?        yang:date-and-time
```

```
        +--rw dependencies
           +--rw dependency* [type id]
              +--rw type               -> /subservices/subservice/type
              +--rw id                 -> /subservices/
subservice[type=current()/../type]/id
              +--rw dependency-type?   identityref
```

7.3. Concepts

Under some circumstances, vendor-specific subservice types might be required. As an example of this vendor-specific implementation, this section shows how to augment the ietf-service-assurance-device module

to add support for the subservice DeviceHealthy, specific to the ACME Corporation. The new parameter is acme-specific-parameter.

7.4. YANG Module

```
module example-service-assurance-device-acme {
  yang-version 1.1;
  namespace "urn:example:example-service-assurance-device-acme";
  prefix example-service-assurance-device-acme;

  import ietf-service-assurance {
    prefix "service-assurance";
  }

  import ietf-service-assurance-device {
    prefix "service-assurance-device";
  }

  organization
    "IETF NETCONF (Network Configuration) Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/netconf/>
     WG List:  <mailto:netconf@ietf.org>
     Author:   Benoit Claise  <mailto:bclaise@cisco.com>
     Author:   Jean Quilbeuf   <mailto:jquilbeu@cisco.com>";
  description
   "This module extends the ietf-service-assurance-device module to add
    support for the subservice DeviceHealthy, specific to the ACME Corporation.

    ACME Network Device is healthy.

    The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
    'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
    'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document
    are to be interpreted as described in BCP 14 (RFC 2119)
    (RFC 8174) when, and only when, they appear in all
    capitals, as shown here.

    Copyright (c) 2019 IETF Trust and the persons identified as
    authors of the code.  All rights reserved.

    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD License
    set forth in Section 4.c of the IETF Trust's Legal Provisions
    Relating to IETF Documents
    (https://trustee.ietf.org/license-info).
```

```
    This version of this YANG module is part of RFC XXXX; see the
    RFC itself for full legal notices.  ";

  revision 2020-01-13 {
    description
      "Added the maintenance window concept.";
    reference
      "RFC xxxx: Title to be completed";
  }

  revision 2019-11-16 {
    description
      "Initial revision.";
    reference
        "RFC xxxx: Title to be completed";
  }

  identity device-acme-idty {
    base service-assurance-device:device-idty;
    description "Network Device is healthy.";
  }

  augment /service-assurance:subservices/service-assurance:subservice/service-
assurance:parameter {
       description
         "Specify the required parameters for a new subservice type";
       container acme-device-idty{
           when "derived-from-or-self(../service-assurance:type, 'device-acme-
idty')";
           description
             "Specify the required parameters for the device-acme-idty
subservice type";

           leaf device {
               type string;
               description "The device to monitor.";
           }

           leaf acme-specific-parameter {
               type string;
               description "The ACME Corporation sepcific parameter.";
           }
       }
  }
}
```

8. Security Considerations

The YANG module specified in this document defines a schema for data

that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer

is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/ creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

- /subservices/subservice/type
- /subservices/subservice/id
- /subservices/subservice/under-maintenance
- /subservices/subservice/maintenance-contact

9. IANA Considerations

9.1. The IETF XML Registry

This document registers two URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested:

URI: urn:ietf:params:xml:ns:yang:ietf-service-assurance
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-service-assurance-device
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-service-assurance-interface
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

9.2. The YANG Module Names Registry

This document registers three YANG modules in the YANG Module Names registry [RFC7950]. Following the format in [RFC7950], the the following registrations are requested:

```
name:       ietf-service-assurance
namespace:  urn:ietf:params:xml:ns:yang:ietf-service-assurance
prefix:     inc
reference:  RFC XXXX

name:       ietf-service-assurance-device
namespace:  urn:ietf:params:xml:ns:yang:ietf-service-assurance-device
prefix:     inc
reference:  RFC XXXX

name:       ietf-service-assurance-interface
namespace:  urn:ietf:params:xml:ns:yang:ietf-service-assurance-interface
prefix:     inc
reference:  RFC XXXX
```

10. Open Issues

-None

11. References

11.1. Normative References

[draft-claise-opsawg-service-assurance-architecture] Claise, B. and J. Quilbeuf, "draft-claise-opsawg-service-assurance-architecture", 2020.

[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>.

[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>.

[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>.

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

11.2. Informative References

[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>.

[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>.

[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>.

Appendix A. Changes between revisions

v04 - v05

- Added the concept of symptoms-history-start
- Changed label to description, under symptoms. This was confusing as there was two labels in the models

v03 - v04

- Add the interface subservice, with two parameters

v02 - v03

- Added the maintenace window concepts

v01 - v02

- Improved leaf naming
- Clarified some concepts: symptoms, dependency

v00 - v01

- Terminology clarifications
- Provide example of impacting versus impacted dependencies

Acknowledgements

The authors would like to thank Jan Lindblad for his help during the design of these YANG modules. The authors would like to thank Stephane Litkowski and Charles Eckel for their reviews.

Authors' Addresses

Benoit Claise
Huawei

Email: benoit.claise@huawei.com

Jean Quilbeuf
Independent

Email: jean@quilbeuf.net

Paolo Lucente
NTT
Siriusdreef 70-72
Hoofddorp, WT 2132
Netherlands

Email: paolo@ntt.net

Paolo Fasano
TIM S.p.A
via G. Reiss Romoli, 274
10148 Torino
Italy

Email: paolo2.fasano@telecomitalia.it

Thangam Arumugam
Cisco Systems, Inc.
Milpitas (California)
United States

Email: tarumuga@cisco.com