Workgroup: OPSAWG
Internet-Draft:
draft-ietf-opsawg-service-assurance-yang-05
Published: 29 April 2022
Intended Status: Standards Track
Expires: 31 October 2022
Authors: B. Claise J. Quilbeuf P. Lucente P. Fasano
Huawei Huawei NTT TIM S.p.A
T. Arumugam
Cisco Systems, Inc.

YANG Modules for Service Assurance

Abstract

This document specifies YANG modules representing assurance graphs. These graphs represent the assurance of a given service by decomposing it into atomic assurance elements called subservices. A companion RFC, Service Assurance for Intent-based Networking Architecture, presents an architecture for implementing the assurance of such services.

The YANG data models in this document conforms to the Network Management Datastore Architecture (NMDA) defined in RFC 8342.

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 31 October 2022.

Table of Contents

1. Introduction

The "Service Assurance for Intent-based Networking Architecture" [I-D.ietf-opsawg-service-assurance-architecture], specifies the architecture 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

1.1. 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 13 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The terms used in this document are defined in [I-D.ietf-opsawg-service-assurance-architecture]

2. 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 main module represents the configuration (subservice and dependencies) and operational data (health status and symptoms) in a single tree. Other modules follows the same pattern. Thus, the modules presented in this document conform to the Network Management Datastore Architecture defined in [RFC8342].

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

The third YANG module, ietf-service-assurance-interface, is another example that extends the ietf-service-assurance module. This extension adds support for the interface subservice.

We provide additional examples in the appendix. The module example-service-assurance-device-acme extends the ietf-service-assurance-device module to customize it for devices of the fictional ACME Corporation. Additional vendor-specific parameters might be added the same way. We also provide the modules example-service-assurance-ip-connectivity and example-service-assurance-is-is to completely model the example from the SAIN architecture draft [I-D.ietf-opsawg-service-assurance-architecture].

3. Base ietf-service-assurance YANG module

3.1. 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:counter64
  +--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?                       union
        +--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                 leafref
              +--rw dependency-type?   identityref
```

3.2. 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:

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

- ip-connectivity: 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 status of each subservice contains a list of symptoms. 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 relation between the health score and the health-score-weight of the currently active symptoms is not explicitly defined in this draft. The only requirement is that a non-maximal score must be explained by at least one symptom. A way to enforce that requirement is to first detect symptoms and then compute the health score based on the health-score-weight of the detected symptoms. As an example, this computation could be to sum the health-score-weight of the active symptoms, subtract that value from 100 and change the value to 0 if negative. The relation between health-score and health-score-weight is left to the implementor (of an agent [I-D.ietf-opsawg-service-assurance-architecture]). To consider for implementing this relation: the health-score is mostly for humans, the symptoms are what the closed loop automation can build on.

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.

A subservice declaration 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,

* One or more 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 interface subservice, 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 the interface subservice towards the device subservice is "impacting". On the other hand, a dependency towards the ecmp-load subservice, which checks that the load between ECMP 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 to be assured, 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 [I-D.ietf-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.

3.3. YANG Module

<CODE BEGINS> file "ietf-service-assurance@2022-04-07.yang"

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

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

  organization
    "IETF OPSAWG Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
     WG List:  <mailto:opsawg@ietf.org>
     Author:   Benoit Claise  <mailto:benoit.claise@huawei.com>
     Author:   Jean Quilbeuf   <mailto:jean.quilbeu@huawei.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) 2022 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 Revised 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 2022-04-07 {
  description
    "Shorten prefix. Fix copyright.
     Fix module description";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}
revision 2022-01-04 {
  description
    "Explicitely model a missing value";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}
revision 2021-06-28 {
  description
    "Made service-instance parameters mandatory.";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}
revision 2020-01-13 {
  description
    "Added the maintenance window concept.";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}
revision 2019-11-16 {
  description
    "Initial revision.";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}

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. device).";
  }
  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 to add parameters specific to a new dependency-type.
  // For instance, a specific dependency type could keep symptom
  // whose health-score-weight is larger than a given value.
}

leaf assurance-graph-version {
  type yang:counter64;
  config false;
  mandatory true;
  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;
  config false;
  mandatory true;
  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
        "Type of the subservice, for instance device or interface.";
    }
    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,
          'sain:service-instance-idty')";
    description
      "Specify the parameters of a service instance.";
    leaf service {
      type string;
      mandatory true;
      description
        "Name of the service.";
    }
    leaf instance-name {
      type string;
      mandatory true;
      description
```

```
        "Name of the instance for that service.";
    }
  }
  // Other modules can augment their own cases into here
}
leaf health-score {
  type union {
    type uint8 {
      range "0 .. 100";
    }
    type enumeration {
      enum missing {
        value -1;
        description
          "Explictly represent the fact that the health score is
           missing. This could be used when metrics crucial to
           establish the health score are not collected anymore.";
      }
    }
  }
  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>

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

4.1. Tree View

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

```
module: ietf-service-assurance-device

  augment /sain:subservices/sain:subservice/sain:parameter:
    +--rw parameters
       +--rw device    string
```

4.2. 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:counter64
  +--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
        |  +--:(sain-device:parameters)
        |     +--rw sain-device:parameters
        |        +--rw sain-device:device    string
        +--ro health-score?                       union
        +--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                 leafref
              +--rw dependency-type?   identityref
```

4.3. 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 device subservice type. As the name implies, it monitors and reports the device health, along with some symptoms in case of degradation.

For our device subservice definition, the new identity 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 device 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.

4.4. YANG Module

<CODE BEGINS> file "ietf-service-assurance-device@2022-04-07.yang"

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

  import ietf-service-assurance {
    prefix sain;
    reference
      "RFC xxxx: YANG Modules for 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:benoit.claise@huawei.com>
     Author:   Jean Quilbeuf   <mailto:jean.quilbeuf@huawei.com>";
  description
    "This module extends the ietf-service-assurance module to add
     support for the device subservice.

     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) 2022 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 Revised 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 2022-04-07 {
    description
      "Fix mandatory in augment error by moving when clause.
       Shorten prefix. Fix module description.
       Fix module description";
```

```
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }
  revision 2021-06-28 {
    description
      "Renamed the container for parameters.";
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }
  revision 2020-01-13 {
    description
      "Added the maintenance window concept.";
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }
  revision 2019-11-16 {
    description
      "Initial revision.";
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }

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

  augment "/sain:subservices/sain:subservice/sain:parameter" {
    when "derived-from-or-self(sain:type, 'device-idty')";
    description
      "Specify the required parameters for a new subservice type";
    container parameters {
      description
        "Specify the required parameters for the device-idty
         subservice type";
      leaf device {
        type string;
        mandatory true;
        description
          "The device to monitor.";
      }
    }
  }
}
```

<CODE ENDS>

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

5.1. Tree View

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

```
module: ietf-service-assurance-interface

  augment /sain:subservices/sain:subservice/sain:parameter:
    +--rw parameters
       +--rw device       string
       +--rw interface    string
```

5.2. 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:counter64
  +--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
        |  +--:(sain-interface:parameters)
        |  |  +--rw sain-interface:parameters
        |  |     +--rw sain-interface:device       string
        |  |     +--rw sain-interface:interface    string
        |  +--:(sain-device:parameters)
        |     +--rw sain-device:parameters
        |        +--rw sain-device:device    string
        +--ro health-score?                       union
        +--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                 leafref
              +--rw dependency-type?   identityref
```

5.3. Concepts

For our interface 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 interface 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 parameters are specified.

5.4. YANG Module

<CODE BEGINS> file "ietf-service-assurance-interface@2022-04-07.yang"

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

  import ietf-service-assurance {
    prefix sain;
    reference
      "RFC xxxx: YANG Modules for 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:benoit.claise@huawei.com>
     Author:   Jean Quilbeuf   <mailto:jean.quilbeuf@huawei.com>";
  description
    "This module extends the ietf-service-assurance module to add
     support for the interface subservice.

     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) 2022 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 Revised 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 2022-04-07 {
    description
      "Fix mandatory in augment error by moving when clause.
       Shorten prefix. Fix module description.
       Fix module description";
```

```
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }
  revision 2021-06-28 {
    description
      "Regroup parameters in a container.";
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }
  revision 2020-01-13 {
    description
      "Initial revision.";
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }

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

  augment "/sain:subservices/sain:subservice/sain:parameter" {
    when "derived-from-or-self(sain:type, 'interface-idty')";
    description
      "Specify the required parameters for the interface-idty
       subservice type";
    container parameters {
      description
        "Required parameters for the interface-idty subservice
         type";
      leaf device {
        type string;
        mandatory true;
        description
          "Device supporting the interface.";
      }
      leaf interface {
        type string;
        mandatory true;
        description
          "Name of the interface.";
      }
    }
  }
}
```

<CODE ENDS>

6. 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

7. IANA Considerations

7.1. 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.

7.2. 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:     sain
reference:  RFC XXXX

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

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

8. Open Issues

-None

9. References

9.1. Normative References

[I-D.ietf-opsawg-service-assurance-architecture]
Claise, B., Quilbeuf, J., Lopez, D. R., Voyer, D., and T. Arumugam, "Service Assurance for Intent-based Networking Architecture", Work in Progress, Internet-Draft, draft-ietf-opsawg-service-assurance-architecture-03, 7 March

2022, <https://www.ietf.org/archive/id/draft-ietf-opsawg-service-assurance-architecture-03.txt>.

[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>.

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

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

9.2. 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>.

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

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

A.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 /sain:subservices/sain:subservice/sain:parameter:
    +--rw parameters
       +--rw device                    string
       +--rw acme-specific-parameter   string
```

A.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:counter64
  +--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
        |  +--:(sain-device:parameters)
        |  |  +--rw sain-device:parameters
        |  |     +--rw sain-device:device    string
        |  +--:(example-device-acme:parameters)
        |  |  +--rw example-device-acme:parameters
        |  |     +--rw example-device-acme:device
        |  |     |       string
        |  |     +--rw example-device-acme:acme-specific-parameter
        |  |             string
        |  +--:(sain-interface:parameters)
        |     +--rw sain-interface:parameters
        |        +--rw sain-interface:device       string
        |        +--rw sain-interface:interface    string
        +--ro health-score?                           union
        +--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                 leafref
              +--rw dependency-type?   identityref
```

A.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 custom support for the device subservice, specific to the ACME Corporation. The specific version adds a new parameter, named acme-specific-parameter.

A.4. YANG Module

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

  import ietf-service-assurance {
    prefix sain;
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }
  import ietf-service-assurance-device {
    prefix sain-device;
    reference
      "RFC xxxx: YANG Modules for 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:benoit.claise@huawei.com>
     Author:   Jean Quilbeuf   <mailto:jean.quilbeuf@huawei.com>";
  description
    "This module extends the ietf-service-assurance-device module to
     add specific support for devices of 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) 2022 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 Revised 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 2022-04-07 {
```

```
  description
    "Fix mandatory in augment error by moving when clause.
     Shorten prefix.
     Fix module description";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}
revision 2021-06-28 {
  description
    "Renamed the parameters container.";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}
revision 2020-01-13 {
  description
    "Added the maintenance window concept.";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}
revision 2019-11-16 {
  description
    "Initial revision.";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}

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

augment "/sain:subservices/sain:subservice/sain:parameter" {
  when "derived-from-or-self(sain:type, 'device-acme-idty')";
  description
    "Specify the required parameters for a new subservice type";
  container parameters {
    description
      "Specify the required parameters for the device-acme-idty
       subservice type";
    leaf device {
      type string;
      mandatory true;
      description
        "The device to monitor.";
    }
    leaf acme-specific-parameter {
      type string;
      mandatory true;
```

```
        description
          "The ACME Corporation sepcific parameter.";
      }
    }
  }
}
```

Appendix B. Further Extensions: IP Connectivity and IS-IS subservices

In this section, we provide two additional YANG models to completely cover the example from Figure 2 in [I-D.ietf-opsawg-service-assurance-architecture]. The complete normalization of these modules is to be done in future work.

B.1. IP Connectivity Tree View

That subservice represents the unicast connectivity between two IP addresses located on to different devices. Such a subservice could report symptoms such as "No route found". The following tree diagram [RFC8340] provides an overview of the example-service-assurance-ip-connectivity data model.

```
module: example-service-assurance-ip-connectivity

  augment /sain:subservices/sain:subservice/sain:parameter:
    +--rw parameters
       +--rw device1     string
       +--rw address1    inet:ip-address
       +--rw device2     string
       +--rw address2    inet:ip-address
```

To specify the connectivity that we are interested in, we specify two IP addresses and two devices. The subservice assures that the connectivity between IP address 1 on device 1 and IP address 2 on device 2 is healthy.

B.2. IS-IS Tree View

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

```
module: example-service-assurance-is-is

  augment /sain:subservices/sain:subservice/sain:parameter:
    +--rw parameters
       +--rw instance-name    string
```

The parameter of this subservice is the name of the IS-IS instance to assure.

B.3. Global Tree View

The following tree diagram [RFC8340] provides an overview of the ietf-service-assurance, ietf-service-assurance-device, example-

service-assurance-device-acme, example-service-assurance-ip-connectivity and example-service-assurance-is-is data models.

```
module: ietf-service-assurance
  +--ro assurance-graph-version        yang:counter64
  +--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
        |  +--:(example-ip-connectivity:parameters)
        |  |  +--rw example-ip-connectivity:parameters
        |  |     +--rw example-ip-connectivity:device1     string
        |  |     +--rw example-ip-connectivity:address1
        |  |     |       inet:ip-address
        |  |     +--rw example-ip-connectivity:device2     string
        |  |     +--rw example-ip-connectivity:address2
        |  |             inet:ip-address
        |  +--:(example-is-is:parameters)
        |  |  +--rw example-is-is:parameters
        |  |     +--rw example-is-is:instance-name    string
        |  +--:(sain-device:parameters)
        |  |  +--rw sain-device:parameters
        |  |     +--rw sain-device:device    string
        |  +--:(example-device-acme:parameters)
        |  |  +--rw example-device-acme:parameters
        |  |     +--rw example-device-acme:device
        |  |     |       string
        |  |     +--rw example-device-acme:acme-specific-parameter
        |  |             string
        |  +--:(sain-interface:parameters)
        |     +--rw sain-interface:parameters
        |        +--rw sain-interface:device       string
        |        +--rw sain-interface:interface    string
        +--ro health-score?                                union
        +--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                  leafref
      +--rw dependency-type?   identityref
```

B.4. IP Connectivity YANG Module

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

  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-service-assurance {
    prefix sain;
    reference
      "RFC xxxx: YANG Modules for 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:benoit.claise@huawei.com>
     Author:   Jean Quilbeuf   <mailto:jean.quilbeuf@huawei.com>";
  description
    "This example module extends the ietf-service-assurance module to
     add support for the subservice ip-connectivity.

     Checks whether the ip connectivity between two ip addresses
     belonging to two network devices 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) 2022 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 Revised 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 2022-04-07 {
  description
    "Fix mandatory in augment error by moving when clause.
     Shorten prefix. Fix module description.
     Fix module description";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}
revision 2021-06-28 {
  description
    "Initial revision.";
  reference
    "RFC xxxx: YANG Modules for Service Assurance";
}

identity ip-connectivity-idty {
  base sain:subservice-idty;
  description
    "Checks connectivity between two IP addresses.";
}

augment "/sain:subservices/sain:subservice/sain:parameter" {
  when "derived-from-or-self(sain:type, 'ip-connectivity-idty')";
  description
    "Specify the required parameters for the ip-connectivity-idty
     subservice type";
  container parameters {
    description
      "Required parameters for the ip-connectivity-idty
       subservice type";
    leaf device1 {
      type string;
      mandatory true;
      description
        "Device at the first end of the connection.";
    }
    leaf address1 {
      type inet:ip-address;
      mandatory true;
      description
        "Address at the first end of the connection.";
    }
    leaf device2 {
      type string;
      mandatory true;
      description
        "Device at the second end of the connection.";
    }
    leaf address2 {
```

```
        type inet:ip-address;
        mandatory true;
        description
          "Address at the second end of the connection.";
      }
    }
  }
}
```

B.5. IS-IS YANG Module

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

  import ietf-service-assurance {
    prefix sain;
    reference
      "RFC xxxx: YANG Modules for 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:benoit.claise@huawei.com>
     Author:   Jean Quilbeuf  <mailto:jean.quilbeuf@huawei.com>";
  description
    "This module extends the ietf-service-assurance module to
     add support for the subservice is-is.

     Checks whether an IS-IS instance 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) 2022 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 Revised 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 2022-04-07 {
    description
      "Fix mandatory in augment error by moving when clause.
       Shorten prefix. Fix module description.
       Fix module description";
    reference
```

```
      "RFC xxxx: YANG Modules for Service Assurance";
  }
  revision 2021-06-28 {
    description
      "Initial revision.";
    reference
      "RFC xxxx: YANG Modules for Service Assurance";
  }

  identity is-is-idty {
    base sain:subservice-idty;
    description
      "Health of IS-IS routing protocol.";
  }

  augment "/sain:subservices/sain:subservice/sain:parameter" {
    when "derived-from-or-self(sain:type, 'is-is-idty')";
    description
      "Specify the required parameters for a new subservice
       type";
    container parameters {
      description
        "Specify the required parameters for the IS-IS subservice
         type";
      leaf instance-name {
        type string;
        mandatory true;
        description
          "The instance to monitor.";
      }
    }
  }
}
```

Appendix C. Example of YANG instances

This section contains examples of YANG instances that conform to the YANG modules. The validity of these data instances has been checked using yangson. Yangson requires a YANG library [RFC7895] to define the complete model against which the data instance must be validated. We provide in Appendix D the JSON library file, named "ietf-service-assurance-library.json", that we used for validation.

We provide below the contents of the file "example_configuration_instance.json" which contains the configuration data that models the Figure 2 of [I-D.ietf-opsawg-service-assurance-architecture]. The instance can be validated with yangson by using the invocation "yangson -v example_configuration_instance.json ietf-service-assurance-library.json", assuming all the files (YANG and JSON) defined in this draft reside in the current folder.

```
{
  "ietf-service-assurance:subservices": {
    "subservice": [
      {
        "type": "service-instance-idty",
        "id": "simple-tunnel/example",
        "service-instance-parameter": {
          "service": "simple-tunnel",
          "instance-name": "example"
        },
        "dependencies": {
          "dependency": [
            {
              "type": "ietf-service-assurance-interface:interface-idty",
              "id": "interface/peer1/tunnel0",
              "dependency-type": "impacting-dependency"
            },
            {
              "type": "ietf-service-assurance-interface:interface-idty",
              "id": "interface/peer2/tunnel9",
              "dependency-type": "impacting-dependency"
            },
            {
              "type":
       "example-service-assurance-ip-connectivity:ip-connectivity-idty",
              "id": "connectivity/peer1/2001:db8::1/peer2/2001:db8::2",
              "dependency-type": "impacting-dependency"
            }
          ]
        }
      },
      {
        "type":
       "example-service-assurance-ip-connectivity:ip-connectivity-idty",
        "id": "connectivity/peer1/2001:db8::1/peer2/2001:db8::2",
        "example-service-assurance-ip-connectivity:parameters": {
          "device1": "Peer1",
          "address1": "2001:db8::1",
          "device2": "Peer2",
          "address2": "2001:db8::2"
        },
        "dependencies": {
          "dependency": [
            {
              "type": "ietf-service-assurance-interface:interface-idty",
              "id": "interface/peer1/physical0",
              "dependency-type": "impacting-dependency"
            },
            {
```

```
        "type": "ietf-service-assurance-interface:interface-idty",
        "id": "interface/peer2/physical5",
        "dependency-type": "impacting-dependency"
      },
      {
        "type": "example-service-assurance-is-is:is-is-idty",
        "id": "is-is/instance1",
        "dependency-type": "impacting-dependency"
      }
    ]
  }
},
{
  "type": "example-service-assurance-is-is:is-is-idty",
  "id": "is-is/instance1",
  "example-service-assurance-is-is:parameters": {
    "instance-name": "instance1"
  }
},
{
  "type": "ietf-service-assurance-interface:interface-idty",
  "id": "interface/peer1/tunnel0",
  "ietf-service-assurance-interface:parameters": {
    "device": "Peer1",
    "interface": "tunnel0"
  },
  "dependencies": {
    "dependency": [
      {
        "type": "ietf-service-assurance-interface:interface-idty",
        "id": "interface/peer1/physical0",
        "dependency-type": "impacting-dependency"
      }
    ]
  }
},
{
  "type": "ietf-service-assurance-interface:interface-idty",
  "id": "interface/peer1/physical0",
  "ietf-service-assurance-interface:parameters": {
    "device": "Peer1",
    "interface": "physical0"
  },
  "dependencies": {
    "dependency": [
      {
        "type": "ietf-service-assurance-device:device-idty",
        "id": "interface/peer1",
        "dependency-type": "impacting-dependency"
```

```
      }
    ]
  }
},
{
  "type": "ietf-service-assurance-device:device-idty",
  "id": "interface/peer1",
  "ietf-service-assurance-device:parameters": {
    "device": "Peer1"
  }
},
{
  "type": "ietf-service-assurance-interface:interface-idty",
  "id": "interface/peer2/tunnel9",
  "ietf-service-assurance-interface:parameters": {
    "device": "Peer2",
    "interface": "tunnel9"
  },
  "dependencies": {
    "dependency": [
      {
        "type": "ietf-service-assurance-interface:interface-idty",
        "id": "interface/peer2/physical5",
        "dependency-type": "impacting-dependency"
      }
    ]
  }
},
{
  "type": "ietf-service-assurance-interface:interface-idty",
  "id": "interface/peer2/physical5",
  "ietf-service-assurance-interface:parameters": {
    "device": "Peer2",
    "interface": "physical5"
  },
  "dependencies": {
    "dependency": [
      {
        "type": "ietf-service-assurance-device:device-idty",
        "id": "interface/peer2",
        "dependency-type": "impacting-dependency"
      }
    ]
  }
},
{
  "type": "ietf-service-assurance-device:device-idty",
  "id": "interface/peer2",
  "ietf-service-assurance-device:parameters": {
```

```
          "device": "Peer2"
        }
      }
    ]
  }
}
```

Appendix D. YANG Library for Service Assurance

This section provides the JSON encoding of the YANG library [RFC7895] listing all modules defined in this draft and their dependencies. This library can be used to validate data instances using yangson, as explained in the previous section.

```
{
  "ietf-yang-library:modules-state": {
    "module-set-id": "ietf-service-assurance@2022-04-07",
    "module": [
      {
        "name": "ietf-service-assurance",
        "namespace":
          "urn:ietf:params:xml:ns:yang:ietf-service-assurance",
        "revision": "2022-04-07",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-service-assurance-device",
        "namespace":
          "urn:ietf:params:xml:ns:yang:ietf-service-assurance-device",
        "revision": "2022-04-07",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-service-assurance-interface",
        "namespace":
         "urn:ietf:params:xml:ns:yang:ietf-service-assurance-interface",
        "revision": "2022-04-07",
        "conformance-type": "implement"
      },
      {
        "name": "example-service-assurance-device-acme",
        "namespace":
          "urn:example:example-service-assurance-device-acme",
        "revision": "2022-04-07",
        "conformance-type": "implement"
      },
      {
        "name": "example-service-assurance-is-is",
        "namespace": "urn:example:example-service-assurance-is-is",
        "revision": "2022-04-07",
        "conformance-type": "implement"
      },
      {
        "name": "example-service-assurance-ip-connectivity",
        "namespace":
          "urn:example:example-service-assurance-ip-connectivity",
        "revision": "2022-04-07",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-yang-types",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types",
        "revision": "2021-04-14",
```

```
        "conformance-type": "import"
      },
      {
        "name": "ietf-inet-types",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types",
        "revision": "2021-02-22",
        "conformance-type": "import"
      }
    ]
  }
}
```

Appendix E. Changes between revisions

v04 - v05

- *Remove Guidelines section
- *Move informative parts (examples) to appendix
- *Minor text edits and reformulations

v03 - v04

- *Fix YANG errors
- *Change is-is and ip-connectivity subservices from ietf to example.
- *Mention that models are NMDA compliant
- *Fix typos, reformulate for clarity

v02 - v03

- *Change counter32 to counter64 to avoid resetting too frequently
- *Explain why relation between health-score and symptom's health-score-weight is not defined and how it could be defined

v01 - v02

- *Explicitly represent the fact that the health-score could not be computed (value -1)

v00 - v01

- *Added needed subservice to model example from architecture draft
- *Added guideline section for naming models
- *Added data instance examples and validation procedure
- *Added the "parameters" container in the interface YANG module to correct a bug.

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, Charles Eckel, Mohamed Boucadair and Tom Petch for their reviews.

Authors' Addresses

Benoit Claise
Huawei

Email: benoit.claise@huawei.com

Jean Quilbeuf
Huawei

Email: jean.quilbeuf@huawei.com

Paolo Lucente
NTT
Siriusdreef 70-72
2132 Hoofddorp
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