Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 3, 2015

T. Senevirathne
N. Finn
D. Kumar, Ed.
S. Salam
Cisco
Q. Wu, Ed.
M. Wang
Huawei

June 1, 2015

Generic YANG Data Model for Operations, Administration, and Maintenance (OAM)

draft-tissa-lime-yang-oam-model-05

Abstract

This document presents base YANG Data model for OAM. It provides a protocol-independent and technology-independent abstraction of key OAM constructs. Based model presented here can be extended to include technology specific details. Leading to uniformity between OAM technologies and support nested OAM workflows (i.e., performing OAM functions at different layers through a unified interface).

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 http://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 December 3, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

Operations, Administration, and Maintenance (OAM) are important networking functions that allow operators to:

1. Monitor networks (Connectivity Verification, Continuity Check).

2. Troubleshoot failures (Fault verification and isolation).

3. Measure Performance

An overview of OAM tools is presented at [RFC7276].

Ping and Traceroute [RFC792], [RFC4443] are well-known fault verification and isolation tools, respectively, for IP networks. Over the years, different technologies have developed similar tools for similar purposes.

[IEEE802.1Q] Connectivity Fault Management is a well-established OAM standard that is widely adopted for Ethernet networks. ITU-T [Y.1731][Y.1731], MEF Service OAM, MPLS-TP [RFC6371], TRILL [RFC7455][RFC7455] all define OAM methods based on manageability frame work of [IEEE802.1Q] [IEEE802.1Q]CFM.

Given the wide adoption of the underlying OAM concepts defined in [IEEE802.1Q][IEEE802.1Q] CFM, it is a reasonable choice to develop the unified management framework based on those concepts. In this document, we take the [IEEE802.1Q][IEEE802.1Q] CFM model and extend it to a technology independent framework and build the corresponding YANG model accordingly. The YANG model presented in this document is the base model and supports generic continuity check, connectivity verification and path discovery. The generic OAM YANG model is designed such that it can be extended to cover various technologies. Technology dependent nodes and RPC commands are defined in technology specific YANG models, which use and extend the base model defined here. As an example, VXLAN uses source UDP port number for flow entropy, while MPLS [RFC4379] uses IP addresses or the label stack for flow entropy in the hashing for multipath selection. To capture this variation, corresponding YANG models would define the applicable structures as augmentation to the generic base model presented here. This accomplishes three purposes: first it keeps each YANG model smaller and manageable. Second, it allows independent development of corresponding YANG models. Third, implementations can limit support to only the applicable set of YANG models. (e.g. TRILL RBridge may only need to implement Generic OAM model and the TRILL YANG model).

All implementations that follow the YANG framework presented in this document MUST implement the generic OAM YANG model presented here.

The YANG data model presented in this document occurs at the management layer. Encapsulations and state machines may differ according to each OAM protocol. A user who wishes to issues a Ping command or a Traceroute or initiate a performance monitoring session can do so in the same manner regardless of the underlying protocol or technology or specific vendor implementation.

As an example, consider a scenario where an IP ping from device A to Device B failed. Between device A and B there are IEEE 802.1 bridges a,b and c. Let's assume a,b and c are using [IEEE802.1Q] CFM. A user upon detecting the IP layer ping failures may decide to drill down to the Ethernet layer and issue the corresponding fault

verification (LBM) and fault isolation (LTM) tools, using the same API. This ability to go up and down to different layers for troubleshooting is referred to as "nested OAM workflow" and is a useful concept that leads to efficient network troubleshooting and maintenance. The OAM YANG model presented in this document facilitates that without needing changes to the underlying protocols.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

2.1. Terminology

CCM Continuity Check Message [IEEE802.1Q].

ECMP
Equal Cost Multipath.

LBM
Loopback Message [IEEE802.1Q].

MP
Maintenance Point [IEEE802.1Q].

MEP
Maintenance End Point [RFC7174] [IEEE802.1Q] [RFC6371].

MIP
Maintenance Intermediate Point [RFC7174] [IEEE802.1Q] [RFC6371].

MA
Maintenance Association [IEEE802.1Q] [RFC7174].

MD
Maintenance Domain [IEEE802.1Q]

MTV
Multi-destination Tree Verification Message.

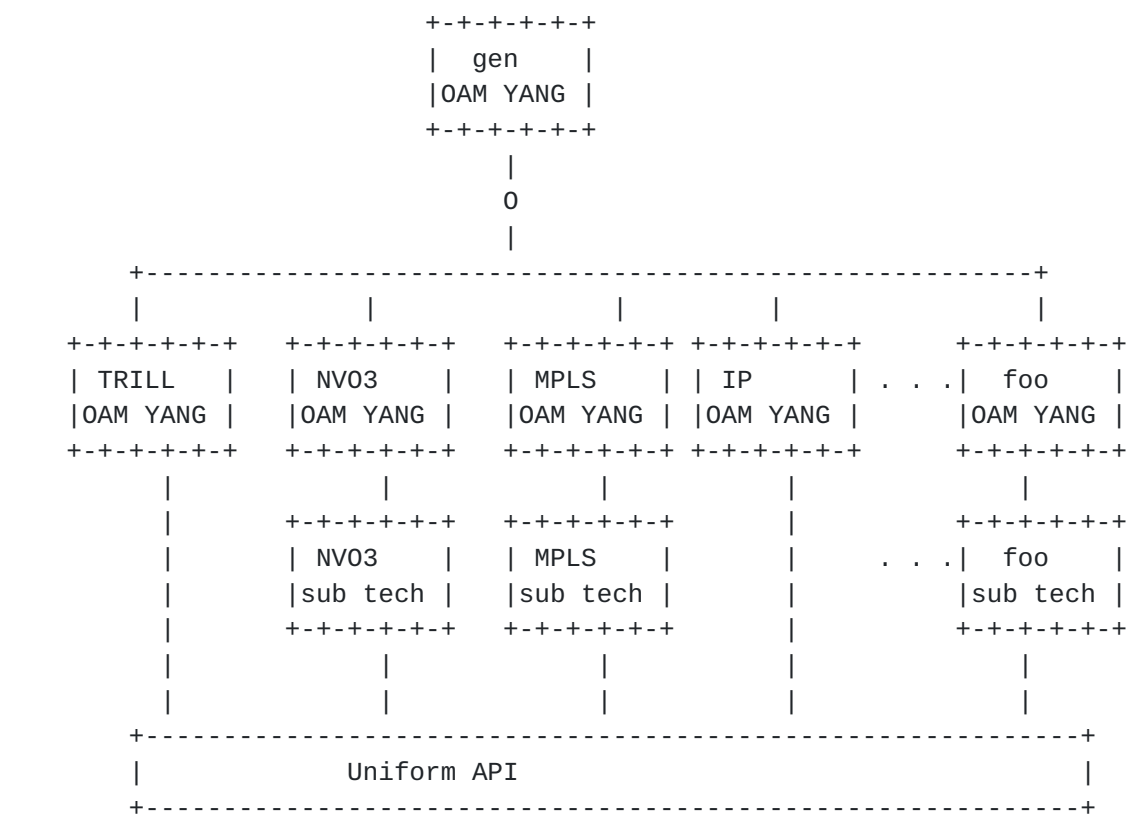
OAM
Operations, Administration, and Maintenance [RFC6291].

TRILL
Transparent Interconnection of Lots of Links [RFC6325].

3. Architecture of Generic YANG Model for OAM

In this document we define a generic YANG model for OAM. The YANG model defined here is generic such that other technologies can extend it for technology specific needs. The Generic OAM YANG model acts as the root for other OAM YANG models. This allows users to traverse between OAM of different technologies at ease through a uniform API set. This is also provides a nested OAM workflow. Figure 1 depicts the relationship of different OAM YANG models to the Generic OAM YANG Model. Some technologies may have different sub-technologies. As an example, consider Network Virtualization Overlays. These could employ either vXLAN or NVGRE as encapsulation. The Generic OAM YANG model provides a framework where technology-specific YANG models can inherit constructs from the base YANG models without needing to redefine them within the sub-technology.

Figure 1 depicts relationship of different YANG modules.



Relationship of OAM YANG model to generic (base) YANG model

4. Overview of the OAM Model

In this document we adopt the concepts of the [IEEE802.1Q] CFM model and structure it such that it can be adapted to different technologies.

At the top of the Model is the Maintenance Domain. Each Maintenance Domain is associated with a Maintenance Name and a Domain Level.

Under each Maintenance Domain there is one or more Maintenance Association (MA). In IP, the MA can be per IP Subnet, in NVO3 this can be per VNI and for TRILL this can be per Fine-Grained Label or for VPLS this can be per VPLS instance.

Under each MA, there can be two or more MEPs (Maintenance End Points). MEPs are addressed by their respective technology specific address identifiers. The YANG model presented here provides flexibility to accommodate different addressing schemes.

In a parallel vertical, presented are the commands. Those, in YANG terms, are the rpc commands. These rpc commands provide uniform APIs

for continuity check,connectivity verification, path discovery and their equivalents as well as other OAM commands.

[IEEE802.1Q] CFM framework requires explicit configuration of OAM entities prior to using any of the OAM tools. Users of Ping and Traceroute tools within IP devices are expecting ability to use OAM tools with no explicit configuration. In order to facilitate zero-touch experience, this document defines a default mode of OAM. The default mode of OAM is referred to as the Base Mode and specifies default values for each of the [IEEE802.1Q] CFM parameters, such as Maintenance Domain Level, Name of the Maintenance Association and Addresses of MEP and so on. The default values of these depend on the technology. Base Mode for TRILL is defined in [RFC7455]. Base mode for other technologies such as NVO3, MPLS and future extensions will be defined in their corresponding documents.

It is important to note that, no specific enhancements are needed in the YANG model to support Base Mode. Implementations that comply with this document, by default implement the data nodes of the applicable technology. Data nodes of the Base Mode are read-only nodes.

4.1. Maintenance Domain (MD) configuration

The container "domains" is the top level container within the gen-oam module. Within the container "domains", separate list is maintained per MD. The MD list uses the key MD-name-string for indexing. MD-name-string is a leaf and derived from type string. Additional name formats as defined in [IEEE802.1Q] or other standards can be included by association of the MD-name-format with an identity-ref. MD-name-format indicates the format of the augmented MD-names. MD-name is presented as choice/case construct. Thus, it is easily augmentable by derivative work.

```
module: ietf-gen-oam
  +--rw domains
     +--rw domain* [technology MD-name-string]
        +--rw technology        identityref
        +--rw MD-name-string    MD-name-string
        +--rw MD-name-format?   identityref
        +--rw (MD-name)?
        |  +--:(MD-name-null)
        |     +--rw MD-name-null?     empty
        +--rw md-level          MD-level  .
```

Snippet of data hierarchy related to OAM domains

4.2. Maintenance Association (MA) configuration

Within a given Maintenance Domain there can be one or more Maintenance Associations (MA). MAs are represented as a list and indexed by the MA-name-string. Similar to MD-name defined previously, additional name formats can be added by augmenting the name-format identity-ref and adding applicable case statements to MA-name.

```
module: ietf-gen-oam
   +--rw domains
      +--rw domain* [technology MD-name-string]
         .
         .
         +--rw MAs
            +--rw MA* [MA-name-string]
               +--rw MA-name-string       MA-name-string
               +--rw MA-name-format?      identityref
               +--rw (MA-name)?
               |  +--:(MA-name-null)
               |     +--rw MA-name-null?        empty
```

Snippet of data hierarchy related to Maintenance Associations (MA)

4.3. Maintenance Endpoint (MEP) configuration

Within a given Maintenance Association (MA), there can be one or more Maintenance End Points (MEP). MEPs are represented as a list within the data hierarchy and indexed by the key MEP-name.

```
module: ietf-gen-oam
   +--rw domains
      +--rw domain* [technology MD-name-string]
         +--rw technology        identityref
         .
         .
         +--rw MAs
            +--rw MA* [MA-name-string]
               +--rw MA-name-string       MA-name-string
               .
               .
               +--rw MEP* [mep-name]
               |  +--rw mep-name             MEP-name
               |  +--rw (MEP-ID)?
               |  |  +--:(MEP-ID-int)
               |  |     +--rw MEP-ID-int?          int32
               |  |  +--:(MEP-ID-tlv)
               |  |     +--rw MEP-ID-type?         int16
               |  |     +--rw MEP-ID-len?          int16
               |  |     +--rw MEP-ID-value?        binary
               |  +--rw MEP-ID-format?       identityref
               |  +--rw (mp-address)?
               |  |  +--:(mac-address)
               |  |  |  +--rw mac-address?   yang:mac-address
               |  |  +--:(ipv4-address)
               |  |  |  +--rw ipv4-address?  inet:ipv4-address
               |  |  +--:(ipv6-address)
               |  |     +--rw ipv6-address?  inet:ipv6-address
      .        .
      .        .
      .        .
```

Snippet of data hierarchy related to Maintenance Endpoint (MEP)

4.4. rpc definitions

The rpc model facilitates issuing commands to a NETCONF server (in this case to the device that need to execute the OAM command) and obtain a response. rpc model defined here abstracts OAM specific commands in a technology independent manner.

There are several rpc commands defined for the purpose of OAM. In this section we present a snippet of the ping command for illustration purposes. Please refer to Section 4 for the complete data hierarchy and Section 5 for the YANG model.

```
module: ietf-gen-oam
   +--rw domains
         +--rw domain* [technology MD-name-string]
         +--rw technology        identityref
   .
   .
rpcs:
   +---x continuity-check
   |  +--ro input
   |  |  +--ro technology             identityref
   |  |  +--ro MD-name-string         MD-name-string
   |  |  +--ro MA-name-string?        MA-name-string
   |  |  +--ro (flow-entropy)?
   |  |  |  +--:(flow-entropy-null)
   |  |  |     +--ro flow-entropy-null?     empty
   |  |  +--ro priority?              uint8
   |  |  +--ro ttl?                   uint8
   |  |  +--ro session-type           enumeration
   |  |  +--ro ecmp-choice?           ecmp-choices
   |  |  +--ro sub-type?              identityref
   |  |  +--ro outgoing-interfaces* [interface]
   |  |  |  +--ro interface          if:interface-ref
   |  |  +--ro source-mep?            MEP-name
   |  |  +--ro destination-mp
   |  |  |  +--ro (mp-address)?
   |  |  |  |  +--:(mac-address)
   |  |  |  |  |  +--ro mac-address?     yang:mac-address
   |  |  |  |  +--:(ipv4-address)
   |  |  |  |  |  +--ro ipv4-address?    inet:ipv4-address
   |  |  |  |  +--:(ipv6-address)
   |  |  |  |     +--ro ipv6-address?    inet:ipv6-address
   |  |  |  +--ro (MEP-ID)?
   |  |  |  |  +--:(MEP-ID-int)
   |  |  |  |     +--ro MEP-ID-int?      int32
   |  |  |  +--ro MEP-ID-format?   identityref
   |  |  +--ro count?                 uint32
   |  |  +--ro interval?              Interval
   |  |  +--ro packet-size?           uint32
   |  +--ro output
   |     +--ro tx-packt-count?    oam-counter32
   |     +--ro rx-packet-count?   oam-counter32
   |     +--ro min-delay?         oam-counter32
   |     +--ro average-delay?     oam-counter32
   |     +--ro max-delay?         oam-counter32
```

Snippet of data hierarchy related to rpc call continuity-check

4.5. OAM data hierarchy

The complete data hierarchy related to the OAM YANG model is presented below. The following notations are used within the data tree and carry the meaning as below.

Each node is printed as:

<status> <flags> <name> <opts> <type>

<status> is one of:

```
+  for current
x  for deprecated
o  for obsolete
```

<flags> is one of:

```
rw for configuration data
ro for non-configuration data
-x for rpcs
-n for notifications
```

<name> is the name of the node

If the node is augmented into the tree from another module, its name is printed as <prefix>:<name>.

<opts> is one of:

```
?  for an optional leaf or choice
!  for a presence container
*  for a leaf-list or list
[<keys>] for a list's keys
```

<type> is the name of the type for leafs and leaf-lists

```
module: ietf-gen-oam
   +--rw domains
      +--rw domain* [technology MD-name-string]
         +--rw technology        identityref
         +--rw MD-name-string    MD-name-string
         +--rw MD-name-format?   identityref
         +--rw (MD-name)?
         |  +--:(MD-name-null)
         |     +--rw MD-name-null?     empty
         +--rw md-level?         MD-level
```

```
            +--rw MAs
               +--rw MA*  [MA-name-string]
                  +--rw MA-name-string       MA-name-string
                  +--rw MA-name-format?      identityref
                  +--rw (MA-name)?
                  |  +--:(MA-name-null)
                  |     +--rw MA-name-null?        empty
                  +--rw (connectivity-context)?
                  |  +--:(context-null)
                  |     +--rw context-null?        empty
                  +--rw mep-direction        MEP-direction
                  +--rw interval?            Interval
                  +--rw loss-threshold?      uint32
                  +--rw ttl?                 uint8
                  +--rw (flow-entropy)?
                  |  +--:(flow-entropy-null)
                  |     +--rw flow-entropy-null?   empty
                  +--rw priority?            uint8
                  +--rw MEP* [mep-name]
                  |  +--rw mep-name             MEP-name
                  |  +--rw (MEP-ID)?
                  |  |  +--:(MEP-ID-int)
                  |  |  |  +--rw MEP-ID-int?          int32
                  |  |  +--:(MEP-ID-tlv)
                  |  |     +--rw MEP-ID-type?         int16
                  |  |     +--rw MEP-ID-len?          int16
                  |  |     +--rw MEP-ID-value?        binary
                  |  +--rw MEP-ID-format?       identityref
                  |  +--rw (mp-address)?
                  |  |  +--:(mac-address)
                  |  |  |  +--rw mac-address?         yang:mac-address
                  |  |  +--:(ipv4-address)
                  |  |  |  +--rw ipv4-address?        inet:ipv4-address
                  |  |  +--:(ipv6-address)
                  |  |     +--rw ipv6-address?        inet:ipv6-address
                  |  +--rw (connectivity-context)?
                  |  |  +--:(context-null)
                  |  |     +--rw context-null?        empty
                  |  +--rw Interface?                 if:interface-ref
                  |  +--rw (topology)?
                  |  |  +--:(topo-null)
                  |  |     +--rw topo-null?           empty
                  |  +--ro admin-status?        leafref
                  |  +--ro oper-status?         leafref
                  |  +--rw (flow-entropy)?
                  |  |  +--:(flow-entropy-null)
                  |  |     +--rw flow-entropy-null?   empty
                  |  +--rw priority?            uint8
```

```
                |  +--rw session* [session-cookie]
                |     +--rw session-cookie             uint32
                |     +--rw ttl?                       uint8
                |     +--rw interval?                  Interval
                |     +--rw enable?                    boolean
                |     +--rw ecmp-choice?               ecmp-choices
                |     +--rw source-mep?                MEP-name
                |     +--rw destination-mep
                |     |  +--rw (MEP-ID)?
                |     |  |  +--:(MEP-ID-int)
                |     |  |  |  +--rw MEP-ID-int?      int32
                |     |  |  +--:(MEP-ID-tlv)
                |     |  |     +--rw MEP-ID-type?     int16
                |     |  |     +--rw MEP-ID-len?      int16
                |     |  |     +--rw MEP-ID-value?    binary
                |     |  +--rw MEP-ID-format?         identityref
                |     +--rw destination-mep-address
                |     |  +--rw (mp-address)?
                |     |     +--:(mac-address)
                |     |     |  +--rw mac-address?    yang:mac-address
                |     |     +--:(ipv4-address)
                |     |     |  +--rw ipv4-address?   inet:ipv4-address
                |     |     +--:(ipv6-address)
                |     |        +--rw ipv6-address?   inet:ipv6-address
                |     +--rw (connectivity-context)?
                |     |  +--:(context-null)
                |     |     +--rw context-null?              empty
                |     +--rw (flow-entropy)?
                |     |  +--:(flow-entropy-null)
                |     |     +--rw flow-entropy-null?         empty
                |     +--rw priority?                  uint8
                |     +--rw outgoing-interface* [interface]
                |        +--rw interface         leafref
                +--rw MIP* [interface]
                |  +--rw interface               if:interface-ref
                +--rw related-oam-layer* [offset]
                   +--rw offset            int32
                   +--rw technology        identityref
                   +--rw MD-name-string    MD-name-string
                   +--rw MA-name-string?   MA-name-string
  rpcs:
     +---x continuity-check
     |  +--ro input
     |  |  +--ro technology            identityref
     |  |  +--ro MD-name-string        MD-name-string
     |  |  +--ro MA-name-string?       MA-name-string
     |  |  +--ro (flow-entropy)?
     |  |  |  +--:(flow-entropy-null)
```

```
|  |  |     +--ro flow-entropy-null?     empty
|  |  +--ro priority?             uint8
|  |  +--ro ttl?                  uint8
|  |  +--ro session-type-enum?    enumeration
|  |  +--ro ecmp-choice?          ecmp-choices
|  |  +--ro sub-type?             identityref
|  |  +--ro outgoing-interfaces* [interface]
|  |  |  +--ro interface          if:interface-ref
|  |  +--ro source-mep?           MEP-name
|  |  +--ro destination-mp
|  |  |  +--ro (mp-address)?
|  |  |  |  +--:(mac-address)
|  |  |  |  |  +--ro mac-address?     yang:mac-address
|  |  |  |  +--:(ipv4-address)
|  |  |  |  |  +--ro ipv4-address?    inet:ipv4-address
|  |  |  |  +--:(ipv6-address)
|  |  |  |     +--ro ipv6-address?    inet:ipv6-address
|  |  |  +--ro (MEP-ID)?
|  |  |  |  +--:(MEP-ID-int)
|  |  |  |  |  +--ro MEP-ID-int?      int32
|  |  |  |  +--:(MEP-ID-tlv)
|  |  |  |     +--ro MEP-ID-type?     int16
|  |  |  |     +--ro MEP-ID-len?      int16
|  |  |  |     +--ro MEP-ID-value?    binary
|  |  |  +--ro MEP-ID-format?      identityref
|  |  +--ro count?                uint32
|  |  +--ro interval?             Interval
|  |  +--ro packet-size?          uint32
|  +--ro output
|     +--ro tx-packt-count?    oam-counter32
|     +--ro rx-packet-count?   oam-counter32
|     +--ro min-delay?         oam-counter32
|     +--ro average-delay?     oam-counter32
|     +--ro max-delay?         oam-counter32
+---x continuity-verification     {connectivity-verification}?
|  +--ro input
|  |  +--ro technology             identityref
|  |  +--ro MD-name-string         MD-name-string
|  |  +--ro MA-name-string?        MA-name-string
|  |  +--ro (flow-entropy)?
|  |  |  +--:(flow-entropy-null)
|  |  |     +--ro flow-entropy-null?     empty
|  |  +--ro priority?             uint8
|  |  +--ro ttl?                  uint8
|  |  +--ro session-type-enum?    enumeration
|  |  +--ro ecmp-choice?          ecmp-choices
|  |  +--ro sub-type?             identityref
|  |  +--ro outgoing-interfaces* [interface]
```

```
      |  |  |  +--ro interface          if:interface-ref
      |  |  +--ro source-mep?           MEP-name
      |  |  +--ro destination-mp
      |  |  |  +--ro (mp-address)?
      |  |  |  |  +--:(mac-address)
      |  |  |  |  |  +--ro mac-address?     yang:mac-address
      |  |  |  |  +--:(ipv4-address)
      |  |  |  |  |  +--ro ipv4-address?    inet:ipv4-address
      |  |  |  |  +--:(ipv6-address)
      |  |  |  |     +--ro ipv6-address?    inet:ipv6-address
      |  |  |  +--ro (MEP-ID)?
      |  |  |  |  +--:(MEP-ID-int)
      |  |  |  |  |  +--ro MEP-ID-int?      int32
      |  |  |  |  +--:(MEP-ID-tlv)
      |  |  |  |     +--ro MEP-ID-type?     int16
      |  |  |  |     +--ro MEP-ID-len?      int16
      |  |  |  |     +--ro MEP-ID-value?    binary
      |  |  |  +--ro MEP-ID-format?      identityref
      |  |  +--ro count?                 uint32
      |  |  +--ro interval?              Interval
      |  |  +--ro packet-size?           uint32
      |  +--ro output
      |     +--ro tx-packt-count?    oam-counter32
      |     +--ro rx-packet-count?   oam-counter32
      |     +--ro min-delay?         oam-counter32
      |     +--ro average-delay?     oam-counter32
      |     +--ro max-delay?         oam-counter32
      +---x path-discovery
         +--ro input
         |  +--ro technology             identityref
         |  +--ro MD-name-string         MD-name-string
         |  +--ro MA-name-string?        MA-name-string
         |  +--ro (flow-entropy)?
         |  |  +--:(flow-entropy-null)
         |  |     +--ro flow-entropy-null?     empty
         |  +--ro priority?              uint8
         |  +--ro ttl?                   uint8
         |  +--ro session-type-enum?     enumeration
         |  +--ro command-sub-type?      identityref
         |  +--ro ecmp-choice?           ecmp-choices
         |  +--ro outgoing-interfaces* [interface]
         |  |  +--ro interface           if:interface-ref
         |  +--ro source-mep?            MEP-name
         |  +--ro destination-mp
         |  |  +--ro (mp-address)?
         |  |  |  +--:(mac-address)
         |  |  |  |  +--ro mac-address?     yang:mac-address
         |  |  |  +--:(ipv4-address)
```

```
         |  |  |  |  +--ro ipv4-address?    inet:ipv4-address
         |  |  |  +--:(ipv6-address)
         |  |  |     +--ro ipv6-address?    inet:ipv6-address
         |  |  +--ro (MEP-ID)?
         |  |  |  +--:(MEP-ID-int)
         |  |  |  |  +--ro MEP-ID-int?      int32
         |  |  |  +--:(MEP-ID-tlv)
         |  |  |     +--ro MEP-ID-type?     int16
         |  |  |     +--ro MEP-ID-len?      int16
         |  |  |     +--ro MEP-ID-value?    binary
         |  |  +--ro MEP-ID-format?      identityref
         |  +--ro count?                 uint32
         |  +--ro interval?              Interval
         +--ro output
            +--ro response* [response-index]
               +--ro response-index     uint8
               +--ro ttl?               uint8
               +--ro destination-mp
               |  +--ro (mp-address)?
               |  |  +--:(mac-address)
               |  |  |  +--ro mac-address?     yang:mac-address
               |  |  +--:(ipv4-address)
               |  |  |  +--ro ipv4-address?    inet:ipv4-address
               |  |  +--:(ipv6-address)
               |  |     +--ro ipv6-address?    inet:ipv6-address
               |  +--ro (MEP-ID)?
               |  |  +--:(MEP-ID-int)
               |  |  |  +--ro MEP-ID-int?      int32
               |  |  +--:(MEP-ID-tlv)
               |  |     +--ro MEP-ID-type?     int16
               |  |     +--ro MEP-ID-len?      int16
               |  |     +--ro MEP-ID-value?    binary
               |  +--ro MEP-ID-format?   identityref
               +--ro tx-packt-count?    oam-counter32
               +--ro rx-packet-count?   oam-counter32
               +--ro min-delay?         oam-counter32
               +--ro average-delay?     oam-counter32
               +--ro max-delay?         oam-counter32
   notifications:
      +---n defect-condition-notification
         +--ro technology          identityref
         +--ro MD-name-string      MD-name-string
         +--ro MA-name-string?     MA-name-string
         +--ro mep-name?           MEP-name
         +--ro defect-type?        identityref
         +--ro generating-mepid
         |  +--ro (MEP-ID)?
         |  |  +--:(MEP-ID-int)
```

```
         |  |  |  +--ro MEP-ID-int?      int32
         |  |  +--:(MEP-ID-tlv)
         |  |     +--ro MEP-ID-type?     int16
         |  |     +--ro MEP-ID-len?      int16
         |  |     +--ro MEP-ID-value?    binary
         |  +--ro MEP-ID-format?         identityref
         +--ro (error)?
            +--:(error-null)
            |  +--ro error-null?         empty
            +--:(error-code)
               +--ro error-code?         int3
               +--ro error-code?         int32
```

data hierarchy of OAM

5. OAM YANG Module

<CODE BEGINS> file "ietf-gen-oam.yang"

```
 module ietf-gen-oam {
namespace "urn:ietf:params:xml:ns:yang:ietf-gen-oam";
prefix goam;

import ietf-interfaces {
  prefix if;
}
import ietf-yang-types {
  prefix yang;
}
import ietf-inet-types {
  prefix inet;
}

organization "IETF LIME Working Group";
contact
  "Tissa Senevirathne tsenevir@cisco.com";
description
  "This YANG module defines the generic configuration,
   statistics and rpc for OAM to be used within IETF in
   a protocol indpendent manner. Functional level
   abstraction is indendent with YANG modeling. It is
   assumed that each protocol maps corresponding
   abstracts to its native format.
   Each protocol may extend the YANG model defined
   here to include protocol specific extensions";

revision 2015-04-09 {
```

```
    description
      "Initial revision. - 04 version";
    reference "draft-tissa-lime-oam";
  }

  /* features */
  feature connectivity-verification {
    description
      "This feature indicates that the server supports
       executing connectivity verification OAM command and
       returning a response. Servers that do not advertise
       this feature will not support executing
       connectivity verification command or rpc model for
       connectivity verification command.";
  }

  /* Identities */

  identity technology-types {
    description
      "this is the base identy of technology types which are
       vpls, nvo3, TRILL, ipv4, ipv6, mpls, etc";
  }

  identity ipv4 {
    base technology-types;
    description
      "technology of ipv4";
  }


  identity ipv6 {
    base technology-types;
    description
      "technology of ipv6";
  }

  identity command-sub-type {
    description
      "defines different rpc command subtypes, e.g rfc792 IP
       ping, rfc4379 LSP ping, rfc6905 trill OAM, this is
       optional for most cases";
  }

  identity icmp-rfc792 {
    base command-sub-type;
    description
      "Defines the command subtypes for ICMP ping";
```

```
    reference "RFC 792";
  }

  identity name-format {
    description
      "This defines the name format, IEEE 8021Q CFM defines varying
      styles of names. It is expected name format as an identity ref
      to be extended with new types.";
  }

  identity name-format-null {
    base name-format;
    description
      "defines name format as null";
  }

  identity identifier-format {
    description
      "identifier-format identity can be augmented to define other
     format identifiers used in MEPD-ID etc";
  }

  identity identifier-format-integer {
    base identifier-format;
    description
      "defines identifier-format to be integer";
  }

  identity defect-types {
    description
      "defines different defect types, e.g. remote rdi,
       mis-connection defect, loss of continuity";
  }

  /* typedefs */
  typedef MEP-direction {
    type enumeration {
      enum "Up" {
        value 0;
        description
          "UP direction.";
      }
      enum "Down" {
        value 1;
        description
          "Down direction.";

      }
```

```
    }
    description
      "MEP direction.";
  }

  typedef MEP-name {
    type string;
    description
      "Generic administrative name for a MEP";
  }

  typedef Interval {
    type uint32;
    units "milliseconds";
    default "1000";
    description
      "Interval between packets in milliseconds.
       0 means no packets are sent.";
  }

  typedef ecmp-choices {
    type enumeration {
      enum "ecmp-use-platform-hash" {
        value 0;
        description
          "Use Platform hashing.";
      }
      enum "ecmp-use-round-robin" {
        value 1;
        description
          "Use round robin hashing.";
      }
    }
    description
      "Equal cost multi Path Choices";
  }

  typedef MD-name-string {
    type string;
    default "";
    description
      "Generic administrative name for an MD";
  }

  typedef MA-name-string {
    type string;
    default "";
    description
```

```
      "Generic administrative name for an MA";
  }

  typedef oam-counter32 {
    type yang:zero-based-counter32;
    description
      "defines 32 bit counter for OAM";
  }

  typedef MD-level {
    type uint32 {
      range "0..255";
    }
    description
      "Maintenance Domain level.  The level may be restricted in
       certain protocols (eg to 0-7)";
  }

  /* groupings */

  grouping topology {
    choice topology {
      case topo-null {
        description
          "this is a placeholder when no topology is needed";
        leaf topo-null {
           type empty;
           description
             "there is no topology define, it will be defined
              in technology specific model.";
        }
      }
      description
        "Topology choices";
    }
    description
      "Topology";
  }

  grouping error-message {
    choice error {
      case error-null {
        description
          "this is a placeholder when no error status is needed";
        leaf error-null {
           type empty;
           description
             "there is no error define, it will be defined in
```

```
              technology specific model.";
        }
      }
      case error-code {
        description
          "this is a placeholder to display error code.";
        leaf error-code {
           type int32;
           description
             "error code is integer value specific to technology.";
        }
      }
      description
        "Error Message choices.";
    }
    description
      "Error Message.";
  }

  grouping mp-address {
    choice mp-address {
      case mac-address {
        leaf mac-address {
          type yang:mac-address;
          description
            "MAC Address";
        }
        description
          "MAC Address based MP Addressing.";
      }
      case ipv4-address {
        leaf ipv4-address {
          type inet:ipv4-address;
          description
            "Ipv4 Address";
        }
        description
          "Ip Address based MP Addressing.";
      }
      case ipv6-address {
        leaf ipv6-address {
          type inet:ipv6-address;
          description
            "Ipv6 Address";
        }
        description
          "ipv6 Address based MP Addressing.";
      }
```

```
      description
        "MP Addressing.";
    }
    description
      "MP Address";
  }

  grouping maintenance-domain-id {
    description
      "Grouping containing leaves sufficient to identify an MD";
    leaf technology {
      type identityref {
        base technology-types;
      }
      mandatory true;

      description
        "Defines the technology";
    }
    leaf MD-name-string {
      type MD-name-string;
      mandatory true;
      description
        "Defines the generic administrative maintenance domain name";
    }
  }

  grouping MD-name {
    leaf MD-name-format {
      type identityref {
        base name-format;
      }
      description
        "Name format.";
    }
    choice MD-name {
      case MD-name-null {
        leaf MD-name-null {
          when "../../../MD-name-format = name-format-null" {
             description
               "MD name format is equal to null format.";
          }
          type empty;
          description
            "MD name Null.";
        }
      }
      description
```

```
        "MD name.";
    }
    description
      "MD name";
  }

  grouping ma-identifier {
    description
      "Grouping containing leaves sufficient to identify an MA";
    leaf MA-name-string {
      type MA-name-string;
      description
        "MA name string.";
    }
  }

  grouping MA-name {
    description
      "MA name";
    leaf MA-name-format {
      type identityref {
        base name-format;
      }
      description
        "Ma name format";
    }
    choice MA-name {
      case MA-name-null {
        leaf MA-name-null {
          when "../../../MA-name-format = name-format-null" {
          description
            "MA";
          }
          type empty;
          description
            "empty";
        }
      }
      description
        "MA name";
    }
  }

  grouping MEP-ID {
    choice MEP-ID {
      default "MEP-ID-int";
      case MEP-ID-int {
        leaf MEP-ID-int {
```

```
          type int32;
          description
            "MEP ID in integer format";
        }
      }
      case MEP-ID-tlv {
         leaf MEP-ID-type {
           type int16;
           description
             "Type of MEP-ID";
         }
         leaf MEP-ID-len {
           type int16;
           description
             "Length of MEP-ID value";
         }
         leaf MEP-ID-value {
            type binary {
              length "12..255";
            }
            description
              "Value please refer RFC6428.";
         }
      }
      description
        "MEP-ID";
    }
    leaf MEP-ID-format {
      type identityref {
        base identifier-format;
      }
      description
        "MEP ID format.";
    }
    description
      "MEP-ID";
  }

  grouping MEP {
    description
      "Defines elements within the MEP";
    leaf mep-name {
      type MEP-name;
      mandatory true;
      description
        "Generic administrative name of the MEP";
    }
    uses MEP-ID;
```

```
    uses mp-address;
    uses connectivity-context;
    leaf Interface {
      type if:interface-ref;
      description
        "Interface name as defined by ietf-interfaces";
    }
    uses topology;
  }

  grouping session-type {
    description
      "This object indicates the current session
       definition.";
    leaf session-type-enum {
      type enumeration {
        enum proactive {
          description
            "The current session is proactive";
        }
        enum on-demand {
          description
            "The current session is on-demand.";
        }
      }
      description
        "session type enum";
    }
  }

  grouping monitor-stats {
    leaf tx-packt-count {
      type oam-counter32;
      description
        "Transmitted Packet count";
    }
    leaf rx-packet-count {
      type oam-counter32;
      description
        "Received packet count";
    }
    leaf min-delay {
      type oam-counter32;
      units milliseconds;
      description
        "Delay is specified in milliseconds";
    }
    leaf average-delay {
```

```
      type oam-counter32;
      units millisecond;
      description
        "average delay in milliseconds";
    }
    leaf max-delay {
      type oam-counter32;
      units millisecond;
      description
        "Maximum delay in milliseconds";
    }
    description
      "Monitor Statistics";
  }

  grouping MIP {
    description
      "defines MIP";
    leaf interface {
      type if:interface-ref;
      description
         "Interface";
    }
  }

  grouping related-oam-layer {
    leaf offset {
      type int32 {
        range "-255..255";
      }
      description
        "defines offset (in MD levels) to a related OAM layer
         +1 is the layer immediately above
         -1 is the layer immediately below";
    }
    uses maintenance-domain-id;
    uses ma-identifier;
    description
      "related OAM layer";
  }

  grouping interface-status {
    description
      "collection of interface related status";
    leaf admin-status {
      type leafref {
        path "/if:interfaces-state/if:interface/if:admin-status";
      }
```

```
      config false;
      description
        "oper status from ietf-interface module";
    }
    leaf oper-status {
      type leafref {
        path "/if:interfaces-state/if:interface/if:oper-status";
      }
      config false;
      description
        "oper status from ietf-interface module";
    }
  }

  grouping connectivity-context {
    description
      "Grouping defining the connectivity context for an MA; for
       example, a VRF for IP, or an LSP for MPLS.  This will be
       augmented by each protocol who use this component";
    choice connectivity-context {
      default "context-null";
      case context-null {
        description
          "this is a place holder when no context is needed";
        leaf context-null {
          type empty;
          description
            "there is no context define";
        }
      }
      description
        "connectivity context";
    }
  }

  grouping priority {
    description
      "Priority used in transmitted packets; for example, in the
       TOS/DSCP field in IP or the Traffic Class field in MPLS";
    leaf priority {
      type uint8;
      description
        "priority";
    }
  }

  grouping flow-entropy {
    description
```

```
      "defines the grouping statement for flow-entropy";
    choice flow-entropy {
      default "flow-entropy-null";
      case flow-entropy-null {
        description
          "this is a place holder when no flow entropy is needed";
        leaf flow-entropy-null {
          type empty;
          description
            "there is no flow entropy defined";
        }
      }
      description
        "Flow entropy";
    }
  }

  grouping measurement-timing-group {
     description
     "This grouping includes objects used for
      proactive and on-demand
      scheduling of PM measurement sessions.";

     container start-time {
        description
        "This container defines the session start time.";
        choice start-time {
          description
          "Measurement sessions tart time can be immediate, relative, or
          absolute.";
          container immediate {
            presence "Start the measurement session immediately.";
            description
            "Start Time of probe immediately.";
          }
          leaf absolute {
            type yang:date-and-time;
            description
            "This objects specifies the scheduled start time
             to perform the on-demand monitoring operations.";
          }
        }
     }

     container stop-time {
        description
        "This container defines the session stop time.";
        choice stop-time {
```

```
          description
          "Measurement session stop time can be none, or absolute.";
          container none {
             presence "Never end the measurement session.";
             description
             "Stop time is never to end.";

          }

          leaf absolute {
            type yang:date-and-time;
            description
              "This objects specifies the scheduled stop time
                to perform the on-demand monitoring operations.";
          }
        }
     }
  }

  container domains {
    description
      "Contains configuration related data. Within the container
       is list of fault domains. Wihin each domian has List of MA.";
    list domain {
      key "technology MD-name-string";
      ordered-by system;
      description
        "Define the list of Domains within the IETF-OAM";
      uses maintenance-domain-id;
      uses MD-name;
      leaf md-level {
        type MD-level;
        description
          "Defines the MD-Level";
      }
      container MAs {
        description
          "This container defines MA, within that have multiple MA
           and within MA have MEP, MIP";
        list MA {
          key "MA-name-string";
          ordered-by system;
          uses ma-identifier;
          uses MA-name;
          uses connectivity-context;
          leaf mep-direction {
            type MEP-direction;
            mandatory true;
```

```
            description
              "Direction for MEPs in this MA";
          }
          leaf interval {
            type Interval;
            default "0";
            description
              "Defines default Keepalive/CC Interval.  May be
               overridden for specific sessions if supported by the
               protocol.";
          }
          leaf loss-threshold {
            type uint32;
            default "3";
            description
              "number of consecutive Keepalive/CC messages missed
               before declaring loss of continuity fault. This is
               monitored per each remote MEP session";
          }
          leaf ttl {
            type uint8;
            default "255";
            description
              "Time to Live";
          }
          uses flow-entropy {
            description
              "Default flow entropy in this MA, which may be
               overridden for particular MEPs, sessions or
               operations";
          }
          uses priority {
            description
              "Default priority for this MA, which may be overridden
               for particular MEPs, sessions or operations.";
          }
          list MEP {
            key "mep-name";
            ordered-by system;
            description
              "contain list of MEPS";
            uses MEP;
            uses interface-status {
              description
                "status of associated interface";
            }
            uses flow-entropy;
            uses priority;
```

```
            list session {
              key "session-cookie";
              ordered-by user;
              description
                "Monitoring session to/from a particular remote MEP.
                 Depending on the protocol, this could represent CC
                 messages received from a single remote MEP (if the
                 protocol uses multicast CCs) or a target to which
                 unicast echo request CCs are sent and from which
                 responses are received (if the protocol uses a
                 unicast request/response mechanism).";
              leaf session-cookie {
                type uint32;
                description
                  "Cookie to identify different sessions, when there
                   are multiple remote MEPs or multiple sessions to
                   the same remote MEP.";
              }
              leaf ttl {
                type uint8;
                default "255";
                description
                  "Time to Live.";
              }
              leaf interval {
                type Interval;
                description
                  "Transmission interval for CC packets for this
                   session.";
              }
              leaf enable {
                type boolean;
                default "false";
                description
                  "enable or disable a monitor session";
              }
              leaf ecmp-choice {
                type ecmp-choices;
                description
                  "0 means use the specified interface
                   1 means use round robin";
              }
              leaf source-mep {
                type MEP-name;
                description
                  "Source MEP for this session, if applicable";
              }
              container destination-mep {
```

```
              uses MEP-ID;
              description
                 "Destination MEP";
            }
            container destination-mep-address {
              uses mp-address;
              description
                 "Destination MEP Address";
            }
            uses connectivity-context;
            uses flow-entropy;
            uses priority;
            list outgoing-interface {
              key "interface";
              leaf interface {
                type leafref {
                  path "/if:interfaces/if:interface/if:name";
                }
                description
                  "Outgoing Interface";
              }
              description
                "outgoing interfaces";
            }
          }
        }
        list MIP {
          key "interface";
          uses MIP;
          description
            "Maintenance Intermediate Point";
        }
        list related-oam-layer {
          key "offset";
          description
            "List of OAM layers above and below that are related to
             current MA. This allow users to easily navigate up and
             down to efficiently troubleshoot a connectivity
             issue";
          uses related-oam-layer;
        }
        description
           "Maintenance Association list";
      }
    }
  }
}
```

```
  notification defect-condition-notification {
    description
      "When defect condition is met this notificiation is sent";
   uses maintenance-domain-id {
      description
        "defines the MD (Maintenance Domain) identifier, which is the
         Generic MD-name-string and the technology.";
    }
    uses ma-identifier;
    leaf mep-name {
      type MEP-name;
      description
        "Indicate which MEP is seeing the error";
    }
    leaf defect-type {
      type identityref {
        base defect-types;
      }
      description
        "The currently active defects on the specific MEP.";
    }
    container generating-mepid {
      uses MEP-ID;
      description
        "Who is generating the error (if known) if
         unknown make it 0.";
    }
    uses error-message {
      description
         "Error message to indicate more details.";
    }
  }
  rpc continuity-check {
    description
      "Generates continuity-check as per RFC7276 Table 4.";
    input {
      uses maintenance-domain-id {
        description
          "defines the MD (Maintenance Domain) identifier, which is
           the generic
           MD-name-string and the technology.";
      }
      uses ma-identifier {
        description
          "identfies the Maintenance association";
      }
      uses flow-entropy;
      uses priority;
```

```
      leaf ttl {
        type uint8;
        default "255";
        description
          "Time to Live";
      }
      uses session-type;
      leaf ecmp-choice {
        type ecmp-choices;
        description
          "0 means use the specified interface
           1 means use round robin";
      }
      leaf sub-type {
        type identityref {
          base command-sub-type;
        }
        description
          "defines different command types";
      }
      list outgoing-interfaces {
        key "interface";
        leaf interface {
          type if:interface-ref;
          description
            "outgoing interface";
        }
        description
          "outgoing Interfaces";
      }
      leaf source-mep {
        type MEP-name;
        description
          "Source MEP";
      }
      container destination-mp {
        uses mp-address;
        uses MEP-ID {
          description "Only applicable if the destination is a MEP";
        }
        description
          "Destination MEP";
      }
      leaf count {
        type uint32;
        default "3";
        description
```

```
          "Number of ping echo request message to send";
      }
      leaf interval {
        type Interval;
        description
          "Interval between echo requests";
      }
      leaf packet-size {
        type uint32 {
          range "64..10000";
        }
        default "64";
        description
          "Size of ping echo request packets, in octets";
      }
    }
    output {
      uses monitor-stats {
        description
          "Stats of continuity check is same as that of
           monitor sessions";
      }
    }
  }

  rpc continuity-verification {
    if-feature connectivity-verification;
    description
      "Generates continuity-verification as per RFC7276 Table 4.";
    input {
      uses maintenance-domain-id {
        description
          "defines the MD (Maintenance Domain) identifier, which is
           the generic
           MD-name-string and the technology.";
      }
      uses ma-identifier {
        description
          "identfies the Maintenance association";
      }
      uses flow-entropy;
      uses priority;
      leaf ttl {
        type uint8;
        default "255";
        description
          "Time to Live";
      }
```

```
      uses session-type;
      leaf ecmp-choice {
        type ecmp-choices;
        description
          "0 means use the specified interface
           1 means use round robin";
      }
      leaf sub-type {
        type identityref {
          base command-sub-type;
        }
        description
          "defines different command types";
      }
      list outgoing-interfaces {
        key "interface";
        leaf interface {
          type if:interface-ref;
          description
            "outgoing interface";
        }
        description
          "outgoing Interfaces";
      }
      leaf source-mep {
        type MEP-name;
        description
          "Source MEP";
      }
      container destination-mp {
        uses mp-address;
        uses MEP-ID {
          description "Only applicable if the destination is a MEP";
        }
        description
          "Destination MEP";
      }
      leaf count {
        type uint32;
        default "3";
        description
          "Number of ping echo request message to send";
      }
      leaf interval {
        type Interval;
        description
          "Interval between echo requests";
      }
```

```
      leaf packet-size {
        type uint32 {
          range "64..10000";
        }
        default "64";
        description
          "Size of ping echo request packets, in octets";
      }
    }
    output {
      uses monitor-stats {
        description
          "Stats of continuity check is same as that of
           monitor sessions";
      }
    }
  }
  rpc path-discovery {
    description
      "Generates Trace-route or Path Trace and return response.
       Referencing RFC7276 for common Toolset name, for IP it's
       Traceroute, for MPLS OAM it's Traceroute mode, for
       MPLS-TP OAM it's Route Tracing, for Pseudowire OAM it's
       LSP Ping, and for TRILL OAM It's Path Tracing tool.
       Starts with TTL
       of one and increment by one at each hop. Untill destination
       reached or TTL reach max value";
    input {
      uses maintenance-domain-id {
        description
          "defines the MD (Maintenance Domain) identifier, which is
           the generic MD-name-string and the technology.";
      }
      uses ma-identifier {
        description
          "identfies the Maintenance association";
      }
      uses flow-entropy;
      uses priority;
      leaf ttl {
        type uint8;
        default "255";
        description
          "Time to Live";
      }
      uses session-type;
      leaf command-sub-type {
        type identityref {
```

```
          base command-sub-type;
        }
        description
          "defines different command types";
      }
      leaf ecmp-choice {
        type ecmp-choices;
        description
          "0 means use the specified interface
           1 means use round robin";
      }
      list outgoing-interfaces {
        key "interface";
        leaf interface {
          type if:interface-ref;
          description
            "Interface.";
        }
        description
          "Outgoing interface list.";
      }
      leaf source-mep {
        type MEP-name;
        description
          "Source MEP";
      }
      container destination-mp {
        uses mp-address;
        uses MEP-ID {
          description "Only applicable if the destination is a MEP";
        }
        description
          "Destination MEP";
      }
      leaf count {
        type uint32;
        default "1";
        description
          "Number of traceroute probes to send.  In protocols where a
           separate message is sent at each TTL, this is the number
           of packets to send at each TTL.";
      }
      leaf interval {
        type Interval;
        description
          "Interval between echo requests";
      }
    }
```

```
    output {
      list response {
        key "response-index";
        leaf response-index {
          type uint8;
          description
            "Arbitrary index for the response.  In protocols that
             guarantee there is only a single response at each TTL
             (eg IP Traceroute), the TTL can be used as the response
             index.";
        }
        leaf ttl {
          type uint8;
          description
            "Time to Live";
        }
        description
           "Time to Live";
        container destination-mp {
          description "MP from which the response has been received";
          uses mp-address;
          uses MEP-ID {
            description
              "Only applicable if the destination is a MEP";
          }
        }
        uses monitor-stats {
          description
            "If count is 1, there is a single delay value reported.";
        }
      }
      description
        "List of response.";
    }
  }
}
```

YANG module of OAM

<CODE ENDS>

6. Base Mode

The Base Mode defines default configuration that MUST be present in the devices that comply with this document. Base Mode allows users to have "zero-touch" experience. Several parameters require technology specific definition.

6.1. MEP Address

In the Base Mode of operation, the MEP Address is by default the IP address of the interface on which the MEP is located.

6.2. MEP ID for Base Mode

In the Base Mode of operation, each device creates a single UP MEP associated with a virtual OAM port with no physical layer (NULL PHY). The MEPID associated with this MEP is zero (0). The choice of MEP-ID zero is explained below.

MEPID is 2 octet field by default. It is never used on the wire except when using CCM. Ping, traceroute and session monitoring does not use the MEPID on its message header. It is important to have method that can derive MEP ID of base mode in an automatic manner with no user intervention. IP address cannot be directly used for this purpose as the MEP ID is much smaller field. For Base Mode of operation we propose to use MEP ID zero (0) as the default MEP-ID.

CCM packet use MEP-ID on the payload. CCM MUST NOT be used in the Base Mode. Hence CCM MUST be disabled on the Maintenance Association of the Base Mode.

If CCM is required, users MUST configure a separate Maintenance association and assign unique value for the corresponding MEP IDs.

[IEEE802.1Q] CFM defines MEP ID as an unsigned integer in the range 1 to 8191. In this document we propose to extend the range to 0 to 65535. Value 0 is reserved for MEP ID of Base Mode operation and MUST NOT be used for other purposes.

6.3. Maintenance Domain

Default MD-LEVEL is set to 3.

6.4. Maintenance Association

MAID [IEEE802.1Q] has a flexible format and includes two parts: Maintenance Domain Name and Short MA name. In the Based Mode of operation, the value of the Maintenance Domain Name must be the character string "GenericBaseMode" (excluding the quotes "). In Base Mode operation Short MA Name format is set to 2-octet integer format (value 3 in Short MA Format field [IEEE802.1Q]) and Short MA name set to 65532 (0xFFFC).

7. Note

This section will be removed or subject to change in the future if any agreement is reached. As per investigation of RFC7276 for performance Monitoring for Loss and Delay are defined for MPLS OAM(RFC6374[RFC6374]), OWAMP (RFC4656[RFC4656]) and TWAMP (RFC5357[RFC5357]) and TRILL OAM (RFC7456[RFC7456]). In case of Performance Monitoring Statistics are common between these technologies thus generic Yang model for Performance will be worked out through separate draft with Augmentation of Generic LIME model. In case of Other Function, it's technology specific and thus should be dealt in technology specific Yang model instead of Generic Model.

8. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241] [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242] [RFC6242]. The NETCONF access control model [RFC6536] [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which 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.

The vulnerable "config true" subtrees and data nodes are the following:

/goam:domains/goam:domain/

/goam:domains/goam:domain/goam:MAs/goam:MA/

/goam:domains/goam:domain/goam:MAs/goam:MA/goam:MEP

/goam:domains/goam:domain/goam:MAs/goam:MA/goam:MEP/goam:session/

Unauthorized access to any of these lists can adversely affect OAM management system handling of end-to-end OAM and coordination of OAM within underlying network layers This may lead to inconsistent configuration, reporting, and presentation for the OAM mechanisms used to manage the network.

9. IANA Considerations

This document registers a URI in the IETF XML registry [RFC3688] [RFC3688]. Following the format in RFC 3688, the following registration is requested to be made:

URI: urn:ietf:params:xml:ns:yang:ietf-gen-oam

Registrant Contact: The IESG.

XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-gen-oam namespace: urn:ietf:params:xml:ns:yang:ietf-gen-oam
prefix: goam reference: RFC XXXX

10. Acknowledgments

Giles Heron came up with the idea of developing a YANG model as a way of creating a unified OAM API set (interface), work in this document is largely an inspiration of that. Alexander Clemm provided many valuable tips, comments and remarks that helped to refine the YANG model presented in this document.

Carlos Pignataro, David Ball and others participated and contributed to this document.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

[RFC792] Postel, J., "Internet Control Message Protocol", RFC 792, September 1981.

11.2. Informative References

[IEEE802.1Q] "Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks", IEEE Std 802.1Q-2011, August 2011.

[RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures", RFC 4379, February 2006.

[RFC4443] Conta, A., Deering, S., and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, March 2006.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, September 2006.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.

[RFC6291] Andersson, L., van Helvoort, H., Bonica, R., Romascanu, D., and S. Mansfield, "Guidelines for the Use of the "OAM" Acronym in the IETF", BCP 161, RFC 6291, June 2011.

[RFC6325] Perlman, R., Eastlake, D., Dutt, D., Gai, S., and A. Ghanwani, "Routing Bridges (RBridges): Base Protocol Specification", RFC 6325, July 2011.

[RFC6371] Busi, I. and D. Allan, "Operations, Administration, and Maintenance Framework for MPLS-Based Transport Networks", RFC 6371, September 2011.

[RFC6374] Frost, D. and S. Bryant, "Packet Loss and Delay Measurement for MPLS Networks", RFC 6374, September 2011.

[RFC7174] Salam, S., Senevirathne, T., Aldrin, S., and D. Eastlake, "Transparent Interconnection of Lots of Links (TRILL) Operations, Administration, and Maintenance (OAM) Framework", RFC 7174, May 2014.

[RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, June 2014.

[RFC7455] Senevirathne, T., Finn, N., Salam, S., Kumar, D., Eastlake, D., Aldrin, S., and Y. Li, "Transparent Interconnection of Lots of Links (TRILL): Fault Management", RFC 7455, March 2015.

[RFC7456] Mizrahi, T., Senevirathne, T., Salam, S., Kumar, D., and D. Eastlake, "Loss and Delay Measurement in Transparent Interconnection of Lots of Links (TRILL)", RFC 7456, March 2015.

[Y.1731] "OAM functions and mechanisms for Ethernet based networks", ITU-T Recommendation G.8013/Y.1731, 2013.

Authors' Addresses

Tissa Senevirathne
CISCO Systems
375 East Tasman Drive.
San Jose, CA 95134
USA

Phone: 408-853-2291
Email: tsenevir@cisco.com

Norman Finn
CISCO Systems
510 McCarthy Blvd
Milpitas, CA 95035
USA

Email: nfinn@cisco.com

Deepak Kumar (editor)
CISCO Systems
510 McCarthy Blvd
Milpitas, CA 95035
USA

Email: dekumar@cisco.com

Samer Salam
CISCO Systems
595 Burrard St. Suite 2123
Vancouver, BC V7X 1J1
Canada

Email: ssalam@cisco.com

Qin Wu (editor)
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: bill.wu@huawei.com

Michael Wang
Huawei Technologies,Co.,Ltd
101 Software Avenue, Yuhua District
Nanjing 210012
China

Email: wangzitao@huawei.com