Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2017

D. Kumar
Cisco
Q. Wu
M. Wang
Huawei

July 8, 2016

# Generic YANG Data Model for Connection Oriented Operations, Administration, and Maintenance(OAM) protocols
draft-ietf-lime-yang-oam-model-07

## Abstract

This document presents a base YANG Data model for connection oriented OAM protocols. It provides a technology-independent abstraction of key OAM constructs for such protocols. The model presented here can be extended to include technology specific details. This guarantees uniformity in the management of OAM protocols and provides support for nested OAM workflows (i.e., performing OAM functions at different levels 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 January 9, 2017.

## Copyright Notice

Table of Contents

## 1. Introduction

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

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

2. Troubleshoot failures (Fault verification and localization).

3. Monitor Performance

An overview of OAM tools is presented in [RFC7276][RFC7276]. Over the years, many technologies have developed similar tools for fault and performance management.

[IEEE802.1Q] Connectivity Fault Management is a well-established OAM standard that is widely adopted for Ethernet networks. ITU-T [G.8013][G.8013], MEF Service OAM, MPLS-TP [RFC6371], TRILL [RFC7455][RFC7455] all define OAM mechanisms based on the 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 for connection oriented OAM 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 define the corresponding YANG model accordingly. The YANG model presented in this document is the base model for connection oriented OAM protocols and supports generic continuity check, connectivity verification (Loopback) and path discovery (traceroute). The generic YANG model for connection oriented OAM is designed to be extensible to other connection oriented technologies. Technology dependent nodes and remote process call (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 TRILL uses either MAC addresses, the VLAN tag or fine grain label, and/or IP addresses 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 goals: First it keeps each YANG model smaller and more 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 model and the TRILL YANG model).

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

The YANG data model presented in this document is generated at the management layer. Encapsulations and state machines may differ according to each OAM protocol. A user who wishes to issues a Continuity Check command or a Loopback 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 Loopback from device A to Device B fails. 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. Upon detecting the Loopback failures, a user may decide to drill down to the lower level at different segments of the path and issue the corresponding fault verification (LBM) and fault isolation (LTM) tools, using the same API. This ability to drill down to a lower layer of the protocol stack at a specific segment within a path for fault localization and troubleshooting is referred to as "nested OAM workflow". It is a useful concept that leads to efficient network troubleshooting and maintenance workflows. The connection oriented 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]. Lower case uses of these words are not to be interpreted as carrying [RFC2119] significance.

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
     (choice)/:(case) Parentheses enclose choice and case nodes,
         and case nodes are also marked with a colon (":")
<type> is the name of the type for leafs and leaf-lists
```

In this document, these words will appear with that interpretation only when in ALL CAPS.

## 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] (Maintenance association End Point [IEEE802.1Q], MEG End Points [RFC6371]).

MIP - Maintenance Intermediate Point [RFC7174] (Maintenance domain Intermediate Point[IEEE802.1Q], MEG Intermediate Point [RFC6371]).

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

MD - Maintenance Domain [IEEE802.1Q]

MEG - Maintenance Entity Group [RFC6371]

MTV - Multi-destination Tree Verification Message.

OAM - Operations, Administration, and Maintenance [RFC6291].

TRILL - Transparent Interconnection of Lots of Links [RFC6325].

CFM - Connectivity Fault Management [RFC7174] [IEEE802.1Q].

RPC - Remote Process Call.

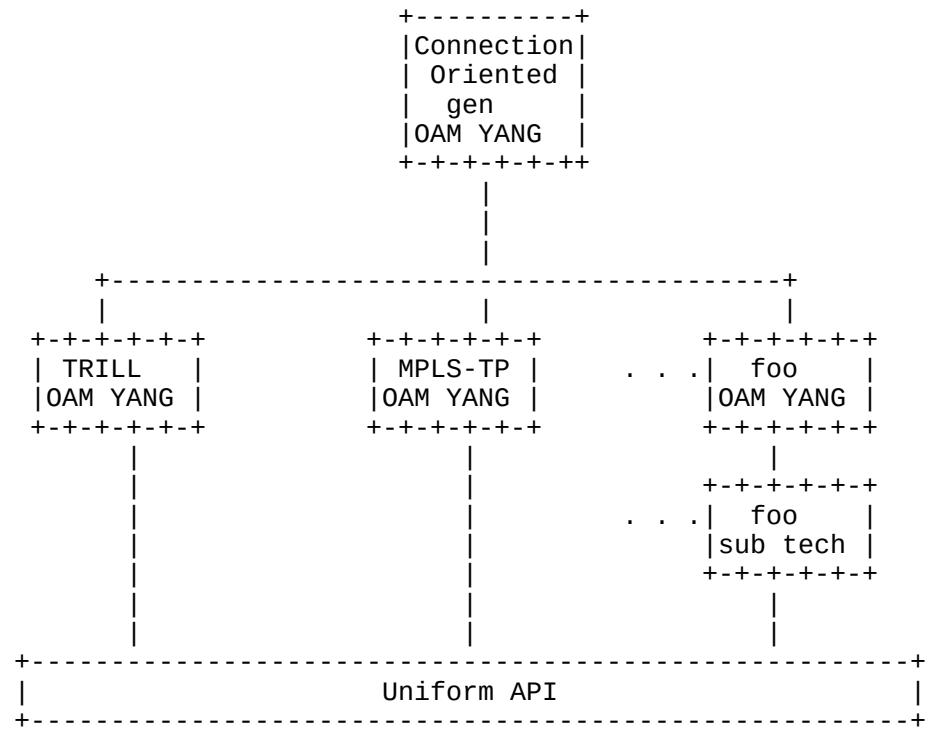
CC - Continuity Check [RFC7276]. Continuity Checks are used to verify that a destination is reachable and therefore also referred to as reachability verification.

CV - Connectivity Verification [RFC7276].Connectivity Verification are used to verify that a destination is connected. It are also referred to as path verification and used to verify not only that the two MPs are connected, but also that they are connected through the expected path, allowing detection of unexpected topology changes.

## 3. Architecture of Generic YANG Model for OAM

In this document we define a generic YANG model for connection oriented OAM protocols. The YANG model defined here is generic in a sense that other technologies can extend it for technology specific needs. The Generic YANG model acts as the root for other OAM YANG models. This allows users to traverse between different OAM protocols with ease through a uniform API set. This also enables a nested OAM workflow. Figure 1 depicts the relationship of different OAM YANG models to the Generic YANG Model for connection oriented OAM. The Generic YANG model for OAM 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 connection oriented OAM protocols.

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 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 Association 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 the vertical direction orthogonal to the Maintenance Domain, presented are the commands. Those, in YANG terms, are the rpc

commands. These rpc commands provide uniform APIs for continuity check, connectivity verification(loopback), path discovery(traceroute) and their equivalents as well as other OAM commands.

The generic YANG model defined here does not require explicit configuration of OAM entities prior to using any of the OAM tools.The OAM tools used here are limited to OAM toolset specified in section 5.1 of [RFC7276]. 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 model parameters, such as Maintenance Domain Level, Name of the Maintenance Association, Addresses of MEPs 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 and future extensions developed in IETF 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-conn-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-conn-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-conn-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
               |  +--rw MEP-ID-format?       identityref
               |  +--rw (mep-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 continuity check command for illustration purposes. Please refer to Section 4 for the complete data hierarchy and Section 5 for the YANG model.

```
module: ietf-conn-oam
   +--rw domains
         +--rw domain* [technology MD-name-string]
         +--rw technology        identityref
   .
   .
```

```
rpcs:
   +---x continuity-check {continuity-check}?
   |  +---w input
   |  |  +---w technology?             identityref
   |  |  +---w MD-name-string          -> /domains/domain/MD-name-string
   |  |  +---w MA-name-string          -> /domains/domain/MAs/MA/MA-name-string
   |  |  +---w cos-id?                 uint8
   |  |  +---w (ttl)?
   |  |  |  +--:(ip-ttl)
   |  |  |  |  +---w ip-ttl?                uint8
   |  |  |  +--:(mpls-ttl)
   |  |  |     +---w mpls-ttl?              uint8
   |  |  +---w sub-type?               identityref
   |  |  +---w source-mep?             -> /domains/domain/MAs/MA/MEP/mep-name
   |  |  +---w destination-mep
   |  |  |  +---w (mep-address)?
   |  |  |  |  +--:(mac-address)
   |  |  |  |  |  +---w mac-address?     yang:mac-address
   |  |  |  |  +--:(ipv4-address)
   |  |  |  |  |  +---w ipv4-address?    inet:ipv4-address
   |  |  |  |  +--:(ipv6-address)
   |  |  |  |     +---w ipv6-address?    inet:ipv6-address
   |  |  |  +---w (MEP-ID)?
   |  |  |  |  +--:(MEP-ID-int)
   |  |  |  |     +---w MEP-ID-int?      int32
   |  |  |  +---w MEP-ID-format?   identityref
   |  |  +---w count?                  uint32
   |  |  +---w cc-transmit-interval?   Interval
   |  |  +---w packet-size?            uint32
   |  +--ro output
   |     +--ro (monitor-stats)?
   |        +--:(monitor-null)
   |           +--ro monitor-null?   empty
   +---x continuity-verification {connectivity-verification}?
   |  +---w input
   |  |  +---w MD-name-string     -> /domains/domain/MD-name-string
   |  |  +---w MA-name-string     -> /domains/domain/MAs/MA/MA-name-string
   |  |  +---w cos-id?            uint8
   |  |  +---w (ttl)?
   |  |  |  +--:(ip-ttl)
   |  |  |  |  +---w ip-ttl?           uint8
   |  |  |  +--:(mpls-ttl)
   |  |  |     +---w mpls-ttl?         uint8
   |  |  +---w sub-type?          identityref
   |  |  +---w source-mep?        -> /domains/domain/MAs/MA/MEP/mep-name
   |  |  +---w destination-mep
   |  |  |  +---w (mep-address)?
   |  |  |  |  +--:(mac-address)
```

```
|  |  |  |  |  +---w mac-address?     yang:mac-address
|  |  |  |  +--:(ipv4-address)
|  |  |  |  |  +---w ipv4-address?    inet:ipv4-address
|  |  |  |  +--:(ipv6-address)
|  |  |  |     +---w ipv6-address?    inet:ipv6-address
|  |  |  +---w (MEP-ID)?
|  |  |  |  +--:(MEP-ID-int)
|  |  |  |     +---w MEP-ID-int?      int32
|  |  |  +---w MEP-ID-format?   identityref
|  |  +---w count?             uint32
|  |  +---w interval?          Interval
|  |  +---w packet-size?       uint32
|  +--ro output
|     +--ro (monitor-stats)?
|        +--:(monitor-null)
|           +--ro monitor-null?   empty
+---x traceroute {traceroute}?
   +---w input
   |  +---w MD-name-string      -> /domains/domain/MD-name-string
   |  +---w MA-name-string      -> /domains/domain/MAs/MA/MA-name-string
   |  +---w cos-id?             uint8
   |  +---w (ttl)?
   |  |  +--:(ip-ttl)
   |  |  |  +---w ip-ttl?             uint8
   |  |  +--:(mpls-ttl)
   |  |     +---w mpls-ttl?           uint8
   |  +---w command-sub-type?   identityref
   |  +---w source-mep?         -> /domains/domain/MAs/MA/MEP/mep-name
   |  +---w destination-mep
   |  |  +---w (mep-address)?
   |  |  |  +--:(mac-address)
   |  |  |  |  +---w mac-address?     yang:mac-address
   |  |  |  +--:(ipv4-address)
   |  |  |  |  +---w ipv4-address?    inet:ipv4-address
   |  |  |  +--:(ipv6-address)
   |  |  |     +---w ipv6-address?    inet:ipv6-address
   |  |  +---w (MEP-ID)?
   |  |  |  +--:(MEP-ID-int)
   |  |  |     +---w MEP-ID-int?      int32
   |  |  +---w MEP-ID-format?   identityref
   |  +---w count?              uint32
   |  +---w interval?           Interval
   +--ro output
      +--ro response* [response-index]
         +--ro response-index     uint8
         +--ro (ttl)?
         |  +--:(ip-ttl)
         |  |  +--ro ip-ttl?            uint8
```

```
            |  +--:(mpls-ttl)
            |     +--ro mpls-ttl?          uint8
            +--ro destination-mep
            |  +--ro (mep-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 (monitor-stats)?
               +--:(monitor-null)
                  +--ro monitor-null?      empty
```

Snippet of data hierarchy related to rpc call continuity-check

## 4.5. notifications

Notification is sent on defect condition and defect clears with Maintenance Domain Name, MA Name, defect-type (The currently active defects), generating-mepid, and defect-message to indicate more details.

## 4.6. monitor statistics

Grouping for monitoring statistics is to be used by Yang modules which Augment Yang to provide statistics due to pro-active OAM like CCM Messages. For example CCM Transmit, CCM Receive, CCM Errors, etc.

## 4.7. OAM data hierarchy

The complete data hierarchy related to the connection oriented OAM YANG model is presented below.

```
module: ietf-conn-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 (MA-ID)?
               |  +--:(MA-id)
               |  |  +--rw MA-id?            uint32
               |  +--:(MEG-ID)
               |     +--rw meg-id?           string
               +--rw (connectivity-context)?
               |  +--:(context-null)
               |     +--rw context-null?     empty
               +--rw cos-id?           uint8
               +--rw cc-enable?        boolean
               +--rw MEP* [mep-name]
               |  +--rw mep-name         MEP-name
               |  +--rw (MEP-ID)?
               |  |  +--:(MEP-ID-int)
               |  |     +--rw MEP-ID-int?      int32
               |  +--rw MEP-ID-format?   identityref
               |  +--rw (mep-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 cos-id?          uint8
               |  +--rw cc-enable?       boolean
               |  +--rw session* [session-cookie]
               |     +--rw session-cookie             uint32
               |     +--rw destination-mep
               |     |  +--rw (MEP-ID)?
               |     |  |  +--:(MEP-ID-int)
               |     |  |     +--rw MEP-ID-int?      int32
               |     |  +--rw MEP-ID-format?   identityref
               |     +--rw destination-mep-address
               |     |  +--rw (mep-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 cos-id?                    uint8
               +--rw MIP* [interface]
                  +--rw interface    if:interface-ref
rpcs:
   +---x continuity-check {continuity-check}?
   |  +---w input
   |  |  +---w technology?             identityref
   |  |  +---w MD-name-string          -> /domains/domain/MD-name-string
   |  |  +---w MA-name-string          -> /domains/domain/MAs/MA/MA-name-string
   |  |  +---w cos-id?                 uint8
   |  |  +---w (ttl)?
   |  |  |  +--:(ip-ttl)
   |  |  |  |  +---w ip-ttl?                uint8
   |  |  |  +--:(mpls-ttl)
   |  |  |     +---w mpls-ttl?              uint8
   |  |  +---w sub-type?               identityref
   |  |  +---w source-mep?             -> /domains/domain/MAs/MA/MEP/mep-name
   |  |  +---w destination-mep
   |  |  |  +---w (mep-address)?
   |  |  |  |  +--:(mac-address)
   |  |  |  |  |  +---w mac-address?     yang:mac-address
   |  |  |  |  +--:(ipv4-address)
   |  |  |  |  |  +---w ipv4-address?    inet:ipv4-address
   |  |  |  |  +--:(ipv6-address)
   |  |  |  |     +---w ipv6-address?    inet:ipv6-address
   |  |  |  +---w (MEP-ID)?
   |  |  |  |  +--:(MEP-ID-int)
   |  |  |  |     +---w MEP-ID-int?      int32
   |  |  |  +---w MEP-ID-format?   identityref
   |  |  +---w count?                  uint32
   |  |  +---w cc-transmit-interval?   Interval
   |  |  +---w packet-size?            uint32
   |  +--ro output
   |     +--ro (monitor-stats)?
   |        +--:(monitor-null)
   |           +--ro monitor-null?   empty
   +---x continuity-verification {connectivity-verification}?
   |  +---w input
   |  |  +---w MD-name-string     -> /domains/domain/MD-name-string
   |  |  +---w MA-name-string     -> /domains/domain/MAs/MA/MA-name-string
   |  |  +---w cos-id?            uint8
   |  |  +---w (ttl)?
   |  |  |  +--:(ip-ttl)
   |  |  |  |  +---w ip-ttl?           uint8
   |  |  |  +--:(mpls-ttl)
   |  |  |     +---w mpls-ttl?         uint8
   |  |  +---w sub-type?          identityref
   |  |  +---w source-mep?        -> /domains/domain/MAs/MA/MEP/mep-name
```

```
   |  |  +---w destination-mep
   |  |  |  +---w (mep-address)?
   |  |  |  |  +--:(mac-address)
   |  |  |  |  |  +---w mac-address?     yang:mac-address
   |  |  |  |  +--:(ipv4-address)
   |  |  |  |  |  +---w ipv4-address?    inet:ipv4-address
   |  |  |  |  +--:(ipv6-address)
   |  |  |  |     +---w ipv6-address?    inet:ipv6-address
   |  |  |  +---w (MEP-ID)?
   |  |  |  |  +--:(MEP-ID-int)
   |  |  |  |     +---w MEP-ID-int?      int32
   |  |  |  +---w MEP-ID-format?   identityref
   |  |  +---w count?             uint32
   |  |  +---w interval?          Interval
   |  |  +---w packet-size?       uint32
   |  +--ro output
   |     +--ro (monitor-stats)?
   |        +--:(monitor-null)
   |           +--ro monitor-null?   empty
   +---x traceroute {traceroute}?
      +---w input
      |  +---w MD-name-string      -> /domains/domain/MD-name-string
      |  +---w MA-name-string      -> /domains/domain/MAs/MA/MA-name-string
      |  +---w cos-id?             uint8
      |  +---w (ttl)?
      |  |  +--:(ip-ttl)
      |  |  |  +---w ip-ttl?             uint8
      |  |  +--:(mpls-ttl)
      |  |     +---w mpls-ttl?           uint8
      |  +---w command-sub-type?   identityref
      |  +---w source-mep?         -> /domains/domain/MAs/MA/MEP/mep-name
      |  +---w destination-mep
      |  |  +---w (mep-address)?
      |  |  |  +--:(mac-address)
      |  |  |  |  +---w mac-address?     yang:mac-address
      |  |  |  +--:(ipv4-address)
      |  |  |  |  +---w ipv4-address?    inet:ipv4-address
      |  |  |  +--:(ipv6-address)
      |  |  |     +---w ipv6-address?    inet:ipv6-address
      |  |  +---w (MEP-ID)?
      |  |  |  +--:(MEP-ID-int)
      |  |  |     +---w MEP-ID-int?      int32
      |  |  +---w MEP-ID-format?   identityref
      |  +---w count?              uint32
      |  +---w interval?           Interval
      +--ro output
         +--ro response* [response-index]
            +--ro response-index     uint8
```

```
            +--ro (ttl)?
            |  +--:(ip-ttl)
            |  |  +--ro ip-ttl?           uint8
            |  +--:(mpls-ttl)
            |     +--ro mpls-ttl?         uint8
            +--ro destination-mep
            |  +--ro (mep-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 (monitor-stats)?
               +--:(monitor-null)
                  +--ro monitor-null?      empty
notifications:
   +---n defect-condition-notification
   |  +--ro technology?         identityref
   |  +--ro MD-name-string      -> /domains/domain/MD-name-string
   |  +--ro MA-name-string      -> /domains/domain/MAs/MA/MA-name-string
   |  +--ro mep-name?           -> /domains/domain/MAs/MA/MEP/mep-name
   |  +--ro defect-type?        identityref
   |  +--ro generating-mepid
   |  |  +--ro (MEP-ID)?
   |  |  |  +--:(MEP-ID-int)
   |  |  |     +--ro MEP-ID-int?      int32
   |  |  +--ro MEP-ID-format?   identityref
   |  +--ro (defect)?
   |     +--:(defect-null)
   |     |  +--ro defect-null?        empty
   |     +--:(defect-code)
   |        +--ro defect-code?        int32
   +---n defect-cleared-notification
      +--ro technology?         identityref
      +--ro MD-name-string      -> /domains/domain/MD-name-string
      +--ro MA-name-string      -> /domains/domain/MAs/MA/MA-name-string
      +--ro mep-name?           -> /domains/domain/MAs/MA/MEP/mep-name
      +--ro defect-type?        identityref
      +--ro generating-mepid
      |  +--ro (MEP-ID)?
      |  |  +--:(MEP-ID-int)
      |  |     +--ro MEP-ID-int?      int32
      |  +--ro MEP-ID-format?   identityref
```

```
      +--ro (defect)?
         +--:(defect-null)
         |  +--ro defect-null?        empty
         +--:(defect-code)
            +--ro defect-code?        int32
```

data hierarchy of OAM

## 5. OAM YANG Module

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

```
module ietf-conn-oam {
  namespace "urn:ietf:params:xml:ns:yang:ietf-conn-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
    "WG Web:    http://tools.ietf.org/wg/lime
     WG List:   mailto:lime@ietf.org
     WG Chair:  Carlos Pignataro cpignata@cisco.com
     WG Chair:  Ron Bonica rbonica@juniper.net
     Editor:    Deepak Kumar dekumar@cisco.com
     Editor:    Qin Wu bill.wu@huawei.com
     Editor:    Zitao Wang wangzitao@huawei.com";
  description
    "This YANG module defines the generic configuration,
  statistics and rpc for connection oriented 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 2016-03-15 {
    description
      "Initial revision. - 05 version";
```

```
    reference "draft-ietf-lime-yang-oam-model";
  }

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

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

  /* Identities */

  identity technology-types {
    description
      "this is the base identy of technology types which are
    TRILL,MPLS-TP,vpls etc";
  }

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

  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 MEP-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";
  }
  identity rdi {
    base defect-types;
    description
      "Indicates the aggregate health of the remote MEPs. ";
  }

  identity remote-mep-defect{
    base defect-types;
    description
      "Indicates that one or more of the remote MEPs is
       reporting a failure ";
  }

  identity loss-of-continuity{
    base defect-types;
    description
    "If no proactive CC-V OAM packets from the source
     MEP (and in the case of CV, this includes the
```

```
     requirement to have the expected globally unique
     Source MEP identifier) are received within the interval
     equal to 3.5 times the receiving MEP's
     configured CC-V reception period. ";
   }

  identity invalid-oam-defect{
    base defect-types;
    description
    "Indicates that one or more invalid OAM messages has been
    received and that 3.5 times that OAM message transmission
    interval has not yet expired.";
  }

  identity cross-connect-defect{
    base defect-types;
    description
    "Indicates that one or more cross-connect defect
    (for example, a service ID does not match the VLAN.)
     messages has been received and that 3.5 times that OAM message
     transmission interval has not yet expired.";
  }

  /* typedefs */


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

  typedef Interval{
    type decimal64{
    fraction-digits 2;
   }
   units "milliseconds";
    description
    "Interval between packets in milliseconds.
    0 means no packets are sent.";
  }

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

```
  typedef MA-name-string {
    type string;
    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 MEG-ID{
    leaf meg-id{
      type string;
      description
       "concatenation of domain and ma, For example a co-routed
        bidirectional LSP, MEG_ID is A1-{Global_ID::Node_ID::
        Tunnel_Num}::Z9-{Global_ID::Node_ID::Tunnel_Num}::LSP_Num.";
    }
    description
      "MEG-ID grouping.";
  }

  grouping time-to-live {
    choice ttl{
      case ip-ttl{
        leaf ip-ttl{
        type uint8;
        default "255";
        description
          "time to live";
        }
      }
     case mpls-ttl{
       leaf mpls-ttl{
```

```
       type uint8;
       description
         "Time to live. When an IP packet is imposed with a label,
        the IP TTL value is first decremented then copied into
        the MPLS TTL. As each LSR the MPLS frame's TTL is
        decremented.This behavior can be modified with no
        mpls ip ttl. When a MPLS label is popped, the MPLS
        TTL value is decremented then copied in the IP TTL
        field. If the MPLS TTL value is great than IP TTL,
        that values is not copied over. This is to prevent
        a possible condition of forwarding loop and TTL
        never reaching 0. When two MPLS labels are swapped,
        decrement by 1 and copy over the result into the new label.
        When a new MPLS labels is pushed, decrement by 1 and copy
        over the result into the new label. When a new MPLS labels
        is popped, decrement by 1 and copy over the result into
        the label below.[RFC3443]";

       }
     }
     description
       "Time to Live.";
   }
   description
     "Time to Live grouping.";
 }
 grouping defect-message {
   choice defect {
     case defect-null {
       description

         "this is a placeholder when no defect status is needed";
       leaf defect-null {
          type empty;
          description
            "there is no defect define, it will be defined in
             technology specific model.";
       }
     }
     case defect-code {
       description
         "this is a placeholder to display defect code.";
       leaf defect-code {
          type int32;
          description
            "defect code is integer value specific to technology.";
       }
     }
```

```
      description
        "defect Message choices.";
    }

    description
      "defect Message.";
  }

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

 grouping monitor-stats {
   description
    "grouping for monitoring statistics, this will be augmented
    by others who use this component";
    choice monitor-stats {

     default "monitor-null";
      case monitor-null {
       description
        "this is a place holder when
         no monitoring statistics is needed";
         leaf monitor-null {
          type empty;
          description
          "there is no monitoring statistics to be defined";
              }
            }
    description
    "define the monitor stats";
          }
        }

  grouping MIP {
    description
      "defines MIP";
```

```
    leaf interface {
      type if:interface-ref;
      description
         "Interface";
    }
  }

  grouping connectivity-context {
    description
      "Grouping defining the connectivity context for an MA; for
       example, a VRF for VPLS, or an LSP for MPLS-TP.  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 cos {
    description

      "Priority used in transmitted packets; for example, in the
       EXP field in MPLS-TP.";
    leaf cos-id {
      type uint8;
      description
        "class of service";
    }
  }

  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;
          choice MA-ID{
               case MA-id{
                leaf MA-id{
                type uint32;
                description
                "MA Identifier";
                }
               description
               "MA ID case";
      }
               case MEG-ID{
                uses MEG-ID;
                description
        "In case MPLS-TP, the MA equivalent to MEG";
                }
               description
      "The MA/MEG identifier";
              }
          uses connectivity-context;
          uses cos {
            description
              "Default class of service for this MA, which may be overridden
               for particular MEPs, sessions or operations.";
          }
             leaf cc-enable{
              type boolean;
              description
              "Indicate whether the CC enable.";
             }
          list MEP {
            key "mep-name";
```

```
            ordered-by system;
            description
            "contain list of MEPS";
            uses MEP;
            uses cos;
                    leaf cc-enable{
                     type boolean;
                     description
                     "Indicate whether the CC enable.";
                    }
            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.";
              }
              container destination-mep {
                uses MEP-ID;
                description
                "Destination MEP";
              }
              container destination-mep-address {
                uses mep-address;
                description
                "Destination MEP Address";
              }
              uses cos;
            }
          }
          list MIP {
            key "interface";
            uses MIP;
            description
            "Maintenance Intermediate Point";
          }
          description
```

```
          "Maintenance Association list";
        }
      }
    }
  }

  notification defect-condition-notification {
    description
      "When defect condition is met this notification is sent";
    leaf technology {
      type identityref {
        base technology-types;
      }
      description
        "the technology";
    }
    leaf MD-name-string {
      type leafref{
           path "/domains/domain/MD-name-string";
          }
      mandatory true;
      description
      "Indicate which MD is seeing the defect";
    }
    leaf MA-name-string{
          type leafref{
           path "/domains/domain/MAs/MA/MA-name-string";
          }
      mandatory true;
      description
      "Indicate which MA is seeing the defect";
        }
    leaf mep-name {
      type leafref{
           path "/domains/domain/MAs/MA/MEP/mep-name";
          }
      description
        "Indicate which MEP is seeing the defect";
    }
    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 defect (if known) if
         unknown make it 0.";
    }
    uses defect-message {
      description
         "defect message to indicate more details.";
    }
  }

  notification defect-cleared-notification {
    description
      "When defect cleared is met this notification is sent";
    leaf technology {
      type identityref {
        base technology-types;
      }
      description
        "the technology";
    }
    leaf MD-name-string {
      type leafref{
           path "/domains/domain/MD-name-string";
         }
      mandatory true;
      description
      "Indicate which MD is seeing the defect";
    }
    leaf MA-name-string{
          type leafref{
           path "/domains/domain/MAs/MA/MA-name-string";
          }
      mandatory true;
      description
      "Indicate which MA is seeing the defect";
        }
    leaf mep-name {
      type leafref{
           path "/domains/domain/MAs/MA/MEP/mep-name";
         }
      description
        "Indicate which MEP is seeing the defect";
    }

    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 defect (if known) if
         unknown make it 0.";
    }
    uses defect-message {
      description
         "defect message to indicate more details.";
    }
  }

  rpc continuity-check {
    if-feature continuity-check;
    description
      "Generates continuity-check as per RFC7276 Table 4.";
    input {
    leaf technology {
      type identityref {
        base technology-types;
      }
      description
        "the technology";
    }
    leaf MD-name-string {
      type leafref{
           path "/domains/domain/MD-name-string";
          }
      mandatory true;
      description
      "Indicate which MD is seeing the defect";
    }
    leaf MA-name-string{
          type leafref{
           path "/domains/domain/MAs/MA/MA-name-string";
          }
      mandatory true;
      description
      "Indicate which MA is seeing the defect";
        }
      uses cos;
      uses time-to-live;
      leaf sub-type {
```

```
        type identityref {
          base command-sub-type;
        }
        description
          "defines different command types";
      }
      leaf source-mep {
       type leafref{
           path "/domains/domain/MAs/MA/MEP/mep-name";
           }
       description
       "Source MEP";
      }
      container destination-mep {
        uses mep-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 continuity-check message to send";
      }
      leaf cc-transmit-interval {
        type Interval;
        description

          "Interval between echo requests";
      }
      leaf packet-size {
        type uint32 {
          range "64..10000";
        }
        default "64";
        description
          "Size of continuity-check packets, in octets";
      }
    }
    output {
      uses monitor-stats {
        description
          "Stats of continuity check.";
      }
```

```
    }
  }

  rpc continuity-verification {
    if-feature connectivity-verification;
    description
      "Generates continuity-verification as per RFC7276 Table 4.";
    input {
     leaf MD-name-string {
      type leafref{
           path "/domains/domain/MD-name-string";
          }
      mandatory true;
      description
      "Indicate which MD is seeing the defect";
     }
     leaf MA-name-string{
          type leafref{
           path "/domains/domain/MAs/MA/MA-name-string";
          }
      mandatory true;
      description
      "Indicate which MA is seeing the defect";
         }
      uses cos;
      uses time-to-live;
      leaf sub-type {

        type identityref {
          base command-sub-type;
        }
        description
          "defines different command types";
      }
      leaf source-mep {
       type leafref{
           path "/domains/domain/MAs/MA/MEP/mep-name";
           }
       description
       "Source MEP";
      }
      container destination-mep {
        uses mep-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 continuity-verification 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 continuity-verification packets, in octets";
      }
    }
    output {
      uses monitor-stats {
        description
          "Stats of continuity check.";
      }
    }
  }
  rpc traceroute {
    if-feature traceroute;
    description
      "Generates Traceroute or Path Trace and return response.
       Referencing RFC7276 for common Toolset name, for
       MPLS-TP OAM it's Route Tracing, 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 {
     leaf MD-name-string {
      type leafref{
           path "/domains/domain/MD-name-string";
          }
      mandatory true;
      description
      "Indicate which MD is seeing the defect";
     }
     leaf MA-name-string{
          type leafref{
```

```
           path "/domains/domain/MAs/MA/MA-name-string";
          }
      mandatory true;
      description
      "Indicate which MA is seeing the defect";
         }
      uses cos;
      uses time-to-live;
      leaf command-sub-type {
        type identityref {

          base command-sub-type;
        }
        description
          "defines different command types";
      }
      leaf source-mep {
       type leafref{
           path "/domains/domain/MAs/MA/MEP/mep-name";
           }
       description
       "Source MEP";
      }
      container destination-mep {
        uses mep-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
             , the TTL can be used as the response
             index.";
        }
        uses time-to-live;
        container destination-mep {
          description "MEP from which the response has been received";
          uses mep-address;
          uses MEP-ID {
            description
              "Only applicable if the destination is a MEP";
          }
        }
        uses monitor-stats {
          description
            "Stats of traceroute.";
        }
      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. 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 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. connection-oriented oam yang model applicability

ietf-conn-oam model defined in this document provides technology-independent abstraction of key OAM constructs for connection oriented protocols. This model can be further extended to include technology specific details, e.g., adding new data nodes with technology specific functions and parameters into proper anchor points of the base model, so as to develop a technology-specific connection-oriented OAM model.

This section demonstrates the usability of the connection-oriented YANG OAM data model to various connection-oriented OAM technologies,

e.g., TRILL and MPLS-TP. Note that, in this section, we only present several snippets of technology-specific model extensions for illustrative purposes. The complete model extensions should be worked on in respective protocol working groups.

### 7.1. Generic YANG Model extension for TRILL OAM

The TRILL YANG module is augmenting connection oriented OAM module for both configuration and RPC commands.

The TRILL YANG module requires the base TRILL module ([I-D.ietf-trill-yang]) to be supported as there is a strong relationship between those modules.

The configuration extensions for connection oriented OAM include MD configuration extension, Technology type extension, MA configuration extension, Connectivity-Context Extension, MEP Configuration Extension, ECMP extension. In the RPC extension, the continuity-check and path-discovery RPC are extended with TRILL specific.

#### 7.1.1. MD Configuration Extension

MD level configuration parameters are management information which can be inherited in the TRILL OAM model and set by connection oriented base model as default values. For example domain name can be set to area-ID in the TRILL OAM case. In addition, at the Maintenance Domain level, domain data node at root level can be augmented with technology type.

Note that MD level configuration parameters provides context information for management system to correlate faults, defects, network failures with location information, which helps quickly identify root causes of network failures.

##### 7.1.1.1. Technology Type Extension

No TRILL technology type has been defined in the connection oriented base model. Therefore a technology type extension is required in the TRILL OAM model. The technology type "trill" is defined as an identity that augments the base "technology-types" defined in the connection oriented base model:

```
identity trill{
 base goam:technology-types;
 description
  "trill type";
}
```

### 7.1.2. MA Configuration Extension

MA level configuration parameters are management information which can be inherited in the TRILL OAM model and set by connection oriented base model as default values. In addition, at the Maintenance Association(MA) level, MA data node at the second level can be augmented with connectivity-context extension.

Note that MA level configuration parameters provides context information for management system to correlate faults, defects, network failures with location information, which helps quickly identify root causes of network failures.

#### 7.1.2.1. Connectivity-Context Extension

In TRILL OAM, one example of connectivity-context is either a 12 bit VLAN ID or a 24 bit Fine Grain Label. The connection oriented base model defines a placeholder for context-id. This allows other technologies to easily augment that to include technology specific extensions. The snippet below depicts an example of augmenting connectivity-context to include either VLAN ID or Fine Grain Label.

```
augment /goam:domains/goam:domain/goam:MAs
/goam:MA /goam:connectivity-context:
      +--:(connectivity-context-vlan)
      |  +--rw connectivity-context-vlan?   vlan
      +--:(connectivity-context-fgl)
         +--rw connectivity-context-fgl?    fgl
```

### 7.1.3. MEP Configuration Extension

The MEP configuration definition in the connection oriented base model already supports configuring the interface of MEP with either MAC address or IP address. In addition, the MEP address can be represented using a 2 octet RBridge Nickname in TRILL OAM . Hence, the TRILL OAM model augments the MEP configuration in base model to add a nickname case into the MEP address choice node as follows:

```
augment /goam:domains/goam:domain/goam:MAs
   /goam:MA/ goam:MEP/goam:mep-address:
         +--:( mep-address-trill)
         |  +--rw mep-address-trill?  tril-rb-nickname
```

In addition, at the Maintenance Association Endpoint(MEP) level, MEP data node at the third level can be augmented with ECMP extension.

#### 7.1.3.1. ECMP Extension

Since TRILL supports ECMP path selection, flow-entropy in TRILL is defined as a 96 octet field in the LIME model extension for TRILL OAM. The snippet below illustrates its extension.

```
 augment /goam:domains/goam:domain/goam:MAs/goam:MA/goam:MEP:
           +--rw flow-entropy-trill?   flow-entropy-trill
   augment /goam:domains/goam:domain/goam:MAs/goam:MA/goam:MEP
   /goam:session:
           +--rw flow-entropy-trill?   flow-entropy-trill
```

### 7.1.4. RPC extension

In the TRILL OAM YANG model, the continuity-check and path-discovery RPC commands are extended with TRILL specific requirements. The snippet below depicts an example of illustrates the TRILL OAM RPC extension.

```
   augment /goam:continuity-check/goam:input:
         +--ro (out-of-band)?
         |  +--:(ipv4-address)
         |  |  +--ro ipv4-address?      inet:ipv4-address
         |  +--:(ipv6-address)
         |  |  +--ro ipv6-address?      inet:ipv6-address
         |  +--:(trill-nickname)
         |     +--ro trill-nickname?    tril-rb-nickname
         +--ro diagnostic-vlan?   boolean
   augment /goam:continuity-check/goam:input:
           +--ro flow-entropy-trill?   flow-entropy-trill
   augment /goam:continuity-check/goam:output:
         +--ro upstream-rbridge?   tril-rb-nickname
         +--ro next-hop-rbridge*   tril-rb-nickname
   augment /goam:path-discovery/goam:input:
         +--ro (out-of-band)?
         |  +--:(ipv4-address)
         |  |  +--ro ipv4-address?      inet:ipv4-address
         |  +--:(ipv6-address)
         |  |  +--ro ipv6-address?      inet:ipv6-address
         |  +--:(trill-nickname)
         |     +--ro trill-nickname?    tril-rb-nickname
         +--ro diagnostic-vlan?   boolean
   augment /goam:path-discovery/goam:input:
           +--ro flow-entropy-trill?   flow-entropy-trill
   augment /goam:path-discovery/goam:output/goam:response:
         +--ro upstream-rbridge?   tril-rb-nickname
         +--ro next-hop-rbridge*   tril-rb-nickname
```

### 7.2. Generic YANG Model extension for MPLS-TP OAM

The MPLS-TP OAM YANG module can augment connection oriented OAM Module with some technology-specific details. And the [mpls-tp-oam-yang] presents the YANG Data model for MPLS-TP OAM.

The configuration extensions for connection oriented OAM include MD configuration extension, Technology type extension, Sub Technology Type Extension ,MA configuration extension, MEP Configuration Extension.

#### 7.2.1. MD Configuration Extension

MD level configuration parameters are management information which can be inherited in the MPLS-TP OAM model and set by LIME base model as default values. For example domain name can be set to area-ID or the provider's Autonomous System Number(ASN) [RFC6370] in the MPLS-TP OAM case. In addition, at the Maintenance Domain level, domain data node at root level can be augmented with technology type and sub-technology type.

Note that MD level configuration parameters provides context information for management system to correlate faults, defects, network failures with location information, which helps quickly identify root causes of network failures

##### 7.2.1.1. Technology Type Extension

No MPLS-TP technology type has been defined in the connection oriented base model, hence it is required in the MPLS OAM model. The technology type "mpls-tp" is defined as an identity that augments the base "technology-types" defined in the connection oriented base model:

```
identity mpls-tp{
      base goam:technology-types;
      description
       "mpls-tp type";
     }
```

##### 7.2.1.2. Sub Technology Type Extension

In MPLS-TP, since different encapsulation types such as IP/UDP Encapsulation, PW-ACH encapsulation can be employed, the "technology-sub-type" data node is defined and added into the MPLS OAM model to further identify the encapsulation types within the MPLS-TP OAM model. Based on it, we also define a technology sub-type for IP/UDP encapsulation and PW-ACH encapsulation. Other Encapsulation types

can be defined in the same way. The snippet below depicts an example of several encapsulation types.

```
identity technology-sub-type {
      description
      "certain implementations can have different
       encapsulation types such as ip/udp, pw-ach and so on.
       Instead of defining separate models for each
       encapsulation, we define a technology sub-type to
    further identify different encapsulations.
    Technology sub-type is associated at the MA level"; }

           identity technology-sub-type-udp {
             base technology-sub-type;
             description
               "technology sub-type is IP/UDP encapsulation";
           }

           identity technology-sub-type-ach {
             base technology-sub-type;
             description
               "technology sub-type is PW-ACH encapsulation";
           }
           }

      augment "/goam:domains/goam:domain/goam:MAs/goam:MA" {
             leaf technology-sub-type {
               type identityref {
                 base technology-sub-type;
               }
             }
           }
```

### 7.2.2. MA Configuration Extension

MA level configuration parameters are management information which can be inherited in the MPLS-TP OAM model and set by Connection Oriented base model as default values. Meg-Id parameter under MA data node will be selected for MPLT-TP OAM model. Therefore one example of MA Name could be MEG LSP ID or MEG Section ID or MEG PW ID[RFC6370].

Note that MA level configuration parameters provides context information for management system to correlate faults, defects, network failures with location information, which helps quickly identify root causes of network failures.

### 7.2.3. MEP Configuration Extension

In MPLS-TP, MEP-ID is either a variable length label value in case of G-ACH encapsulation or a 2 octet unsigned integer value in case of IP/UDP encapsulation. One example of MEP-ID is MPLS-TP LSP_MEP_ID [RFC6370]. In the connection-oriented base model, MEP-ID is defined as a choice/case node which can supports an int32 value, and the same definition can be used for MPLS-TP with no further modification. In addition, at the Maintenance Association Endpoint(MEP) level, MEP data node at the third level can be augmented with Session extension and interface extension.

## 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,Mahesh Jethanandani,Benoit Claise,Ladislav Lhotka,GUBALLA JENS,Yuji Tochio,Gregory Mirsky, Huub van Helvoort, Tom Taylor, Dapeng Liu,Mishael Wexler, Adi Molkho 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, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

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

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://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, <http://www.rfc-editor.org/info/rfc6242>.

[RFC6370] Bocci, M., Swallow, G., and E. Gray, "MPLS Transport Profile (MPLS-TP) Identifiers", RFC 6370, DOI 10.17487/RFC6370, September 2011, <http://www.rfc-editor.org/info/rfc6370>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

## 11.2. Informative References

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

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

[mpls-tp-oam-yang] Zhang, L., Zheng, L., Aldrin, S., and G. Mirsky, "YANG Data Model for MPLS-TP Operations, Administration, and Maintenance", draft-zhang-mpls-tp-yang-oam (work in progress), 2016.

[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, DOI 10.17487/RFC6291, June 2011, <http://www.rfc-editor.org/info/rfc6291>.

[RFC6325] Perlman, R., Eastlake 3rd, D., Dutt, D., Gai, S., and A. Ghanwani, "Routing Bridges (RBridges): Base Protocol Specification", RFC 6325, DOI 10.17487/RFC6325, July 2011, <http://www.rfc-editor.org/info/rfc6325>.

[RFC6371] Busi, I., Ed. and D. Allan, Ed., "Operations, Administration, and Maintenance Framework for MPLS-Based Transport Networks", RFC 6371, DOI 10.17487/RFC6371, September 2011, <http://www.rfc-editor.org/info/rfc6371>.

[RFC7174] Salam, S., Senevirathne, T., Aldrin, S., and D. Eastlake 3rd, "Transparent Interconnection of Lots of Links (TRILL) Operations, Administration, and Maintenance (OAM) Framework", RFC 7174, DOI 10.17487/RFC7174, May 2014, <http://www.rfc-editor.org/info/rfc7174>.

[RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <http://www.rfc-editor.org/info/rfc7276>.

[RFC7455] Senevirathne, T., Finn, N., Salam, S., Kumar, D., Eastlake 3rd, D., Aldrin, S., and Y. Li, "Transparent Interconnection of Lots of Links (TRILL): Fault Management", RFC 7455, DOI 10.17487/RFC7455, March 2015, <http://www.rfc-editor.org/info/rfc7455>.

## Appendix A. Contributors' Addresses

Tissa Senevirathne
Consultant

Email: tsenevir@gmail.com

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

Email: nfinn@cisco.com

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

Email: ssalam@cisco.com

Authors' Addresses

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

Email: dekumar@cisco.com

Qin Wu
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