NETMOD
Internet Draft
Intended status: Standards Track

Tissa Senevirathne
Norman Finn
Deepak Kumar
Samer Salam
Carlos Pignataro
Cisco

June 10, 2014

Expires: December 2014

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

draft-tissa-netmod-oam-01.txt

Status of this Memo

Copyright Notice

Abstract

This document presents YANG Data model for OAM. It provides a protocol-independent and technology-independent abstraction of key OAM constructs. These abstractions span OAM configuration and operational data; they promote uniformity between OAM technologies and support nested OAM workflows (i.e., performing OAM functions at different layers through a unified interface).

Table of Contents

1. Introduction

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

1. Configure networks

2. Monitor networks (Connectivity Verification, Continuity Check)

3. Troubleshoot failures (Fault verification and isolation).

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

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.

[8021Q] Connectivity Fault Management is a well-established OAM standard that is widely adopted for Ethernet networks. ITU-T [Y1731], MEF Service OAM, MPLS-TP [RFC6371], TRILL [TRILLOAMFM] all define OAM methods based on [8021Q] CFM.

Given the wide adoption of the underlying OAM concepts defined in [8021Q] CFM, it is a reasonable choice to develop the unified OAM framework based on those concepts. In this document, we take the [8021Q] 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 IP Ping and Traceroute. 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 unification of OAM, according to the proposal of this document, occurs at the management layer. Encapsulations and state machines may differ according to each 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.

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 [8021Q] CFM. A user upon detecting the IP layer ping failure, 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 RFC-2119 [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 [8021Q]

ECMP - Equal Cost Multipath

LBM - Loopback Message [8021Q]

MP - Maintenance Point [8021Q]

MEP - Maintenance End Point [RFC7174] [8021Q] [RFC6371]

MIP - Maintenance Intermediate Point [RFC7174] [8021Q] [RFC6371]

MA - Maintenance Association [8021Q] [RFC7174]

MD - Maintenance Domain [8021Q]

MTV - Multi-destination Tree Verification Message

OAM - Operations, Administration, and Maintenance [RFC6291]

TRILL - Transparent Interconnection of Lots of Links [RFC6325]

3. Architecture of OAM YANG Model

In this document we define the YANG model for Generic 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 parent YANG models without needing to redefine them within the sub-technology.

Figure 1 depicts relationship of different YANG modules.

```
                          +-+-+-+-+-+
                          |  gen    |
                          |OAM YANG |
                          +-+-+-+-+-+
                               |
                               O
                               |
       +---------------------------------------------------------+
       |              |               |         |                |
   +-+-+-+-+-+   +-+-+-+-+-+   +-+-+-+-+-+ +-+-+-+-+-+      +-+-+-+-+-+
   | TRILL   |   | NVO3    |   | MPLS    | | IP      | . . .|  foo    |
   |OAM YANG |   |OAM YANG |   |OAM YANG | |OAM YANG |      |OAM YANG |
   +-+-+-+-+-+   +-+-+-+-+-+   +-+-+-+-+-+ +-+-+-+-+-+      +-+-+-+-+-+
         |             |             |           |              |
         |       +-+-+-+-+-+   +-+-+-+-+-+ +-+-+-+-+-+      +-+-+-+-+-+
         |       | NVO3    |   | MPLS    | | IP      | . . .|  foo    |
         |       |sub tech |   |sub tech | |OAM YANG |      |sub tec  |
         |       +-+-+-+-+-+   +-+-+-+-+-+ +-+-+-+-+-+      +-+-+-+-+-+
         |             |             |           |              |
         |             |             |           |              |
       +------------------------------------------------------------+
       |             Uniform API                                    |
       +------------------------------------------------------------+
```

Figure 1 Relationship of TRILL OAM YANG model to generic YANG model

4. Overview of the OAM Model

In this document we adopt the concepts of the [8021Q] 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 ping, traceroute and their equivalents as well as other OAM commands.

[8021Q] 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 [8021Q] 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 [TRILLOAMFM]. Section X of this document specifies the Base mode for IP devices. 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 ietf-oam module. Within the container "domains", separate list is maintained per MD. The MD list uses the key MD-name for indexing.

```
module: ietf-oam
   +--rw domains
   |  +--rw domain* [md-name]
   |     +--rw technology        identityref
   |     +--rw md-name-format    MD-name-format
   |     +--rw md-name           binary
   |     +--rw md-level          int32

   .
   .
```

Figure 1 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.

```
module: ietf-oam
   +--rw domains
   |  +--rw domain* [md-name]
   |     +--rw technology        identityref
   |     +--rw md-name-format    MD-name-format
   |     +--rw md-name           binary
   |     +--rw md-level          int32
   |     +--rw MAs!
   |        +--rw MA* [ma-name]
   |           +--rw ma-name-format        MA-name-format
   |           +--rw ma-name               binary
   .
   .
```

Figure 2 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-id.

```
module: ietf-oam
   +--rw domains
   |  +--rw domain* [md-name]
   |     +--rw technology        identityref
   |     +--rw md-name-format    MD-name-format
   |     +--rw md-name           binary
   |     +--rw md-level          int32
   |     +--rw MAs!
   |        +--rw MA* [ma-name]
   |           +--rw ma-name-format        MA-name-format
   |           +--rw ma-name               binary

   .
   .

               +--rw MEP* [mep-id]
               |  +--rw mep-id            MEP-id
               |  +--rw mep-name?         string
               |  +--rw mep-direction    MEP-direction
               |  +--rw ccm-Tx-enable?   boolean
               |  +--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 (context-id)?
               |  |  +--:(context-null)
    .
    .
    .
```

Figure 3 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-oam
      +--rw domains
      |  +--rw Domain*                        [MA-domain-name]
      |     +--rw technology                   technology
      |     +--rw MA-domain-name-format        int32
      |     +--rw MA-domain-name               binary
      |     +--rw MD-level                     int32

      .
      .

      rpcs:
        +---x ping
      |  +--ro input
      |  |  +--ro technology               identityref
      |  |  +--ro md-name-format           MD-name-format
      |  |  +--ro md-name?                 binary
      |  |  +--ro md-level                 int32
      |  |  +--ro ma-name-format           MA-name-format
      |  |  +--ro ma-name                  binary
      |  |  +--ro (context-id)?
      |  |  |  +--:(context-null)
      |  |  |     +--ro context-null?          empty
      |  |  +--ro (flow-entropy)?
      |  |  |  +--:(flow-entropy-null)
      |  |  +--ro ttl?                     uint8
      .
      .
      |  |  +--ro source-mep
      |  |  |  +--ro (mep-address)?
      .
      .
      |  |  |  +--ro mep-id?         MEP-id
      |  |  +--ro destination-mep
      |  |     +--ro (mep-address)?
      .
      .
      |  +--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
```

Figure 4 Snippet of data hierarchy related to rpc call Ping

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: gen-oam
   +--rw domains
      +--rw domain* [md-name technology]
         +--rw technology        identityref
         +--rw md-name-format    MD-name-format
         +--rw md-name           binary
         +--rw md-level          int32
         +--rw MAs!
            +--rw MA* [ma-name]
               +--rw ma-name-format        MA-name-format
               +--rw ma-name               binary
               +--rw (context-id)?
               |  +--:(context-null)
               |     +--rw context-null?          empty
               +--rw ccm-Interval?         CCM-Interval
               +--rw ccm-loss-threshold?   uint32
               +--rw ccm-ttl?              uint8
               +--rw (flow-entropy)?
               |  +--:(flow-entropy-null)
               +--rw MEP* [mep-id]
               |  +--rw mep-id           MEP-id
               |  +--rw mep-name?        string
               |  +--rw mep-direction    MEP-direction
               |  +--rw ccm-Tx-enable?   boolean
               |  +--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 (context-id)?
               |  |  +--:(context-null)
               |  |     +--rw context-null?    empty
               |  +--rw Interface?       if:interface-ref
               |  +--ro admin-status?    leafref
               |  +--ro oper-status?     leafref
               |  +--rw (flow-entropy)?
               |  |  +--:(flow-entropy-null)
               |  +--rw session* [user-cookie destination-mepid]
               |     +--rw user-cookie                uint32
               |     +--rw ttl?                       uint8
               |     +--rw interval?                  uint32
               |     +--rw enable?                    boolean
               |     +--rw ecmp-choice?               ecmp-choices
               |     +--rw destination-mepid          MEP-id
```

```
                  |     +--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
                  |     +--ro ccm-rdi-indicator?         boolean
                  |     +--ro ccm-xcon-count?            oam-counter32
                  |     +--ro ccm-xcon-Indicator?        boolean
                  |     +--rw (context-id)?
                  |     |  +--:(context-null)
                  |     |     +--rw context-null?              empty
                  |     +--rw (flow-entropy)?
                  |     |  +--:(flow-entropy-null)
                  |     +--rw outgoing-interface* [interface]
                  |        +--rw interface    leafref
                  +--rw remote-MEP* [mep-id]
                  |  +--rw mep-id                uint32
                  |  +--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 mep-name?             string
                  |  +--rw ccm-rx-error-count?   oam-counter32
                  +--rw MIP* [interface direction]
                  |  +--rw interface    if:interface-ref
                  |  +--rw direction    MEP-direction
                  +--ro ccm-rdi-indicator?    boolean
                  +--ro ccm-xcon-count?       oam-counter32
                  +--ro ccm-xcon-Indicator?   boolean
                  +--rw nested-oam-layer* [offset]
                     +--rw offset           int8
                     +--rw technology       identityref
                     +--rw md-name-format    MD-name-format
                     +--rw md-name?          binary
                     +--rw md-level          int32
                     +--rw ma-name-format    MA-name-format
                     +--rw ma-name           binary
   rpcs:
      +---x ping
      |  +--ro input
      |  |  +--ro technology             identityref
```

```
      |  |  +--ro md-name-format         MD-name-format
      |  |  +--ro md-name?               binary
      |  |  +--ro md-level               int32
      |  |  +--ro ma-name-format         MA-name-format
      |  |  +--ro ma-name                binary
      |  |  +--ro (context-id)?
      |  |  |  +--:(context-null)
      |  |  |     +--ro context-null?          empty
      |  |  +--ro (flow-entropy)?
      |  |  |  +--:(flow-entropy-null)
      |  |  +--ro ttl?                   uint8
      |  |  +--ro ecmp-choice?           ecmp-choices
      |  |  +--ro sub-type?              identityref
      |  |  +--ro outgoing-interfaces* [interface]
      |  |  |  +--ro interface    if:interface-ref
      |  |  +--ro source-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
      |  |  +--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
      |  +--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 trace-route
         +--ro input
         |  +--ro technology             identityref
         |  +--ro md-name-format         MD-name-format
         |  +--ro md-name?               binary
         |  +--ro md-level               int32
         |  +--ro ma-name-format         MA-name-format
         |  +--ro ma-name                binary
```

```
         |  +--ro (context-id)?
         |  |  +--:(context-null)
         |  |     +--ro context-null?          empty
         |  +--ro (flow-entropy)?
         |  |  +--:(flow-entropy-null)
         |  +--ro ttl?                    uint8
         |  +--ro command-sub-type?      identityref
         |  +--ro ecmp-choice?           ecmp-choices
         |  +--ro outgoing-interfaces* [interface]
         |  |  +--ro interface    if:interface-ref
         |  +--ro source-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
         |  +--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
         +--ro output
            +--ro response* [ttl]
               +--ro 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
               +--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 CCM-RDI-notification
```

```
         +--ro mep-id?          MEP-id
         +--ro remote-mepid?    MEP-id
         +--ro error-message?   string
```

Figure 5 data hierarchy of OAM

6. OAM YANG module

```
   <CODE BEGINS> file "xxx.yang"

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

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

     organization "IETF NETMOD (NETCONF Data Modeling ) 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 protocoal may extend the YANG model defined
        here to include protocol specific extensions";

     revision 2014-04-17 {
       description
         "Initial revision. - 02 version";
       reference "draft-tissa-netmod-oam";
     }

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



     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 ping
          vs udp ping, this is optional for most cases";
     }

     identity icmp-rfc792 {
       base command-sub-type;
       description
         "Defines the command subtypes for ICMP ping";
       reference "RFC 792";
     }

     typedef MEP-direction {
       type enumeration {
         enum "Up" {
           value 0;
         }
         enum "Down" {
           value 1;
         }
       }
     }

     typedef MEP-id {
       type uint32 {
         range "1..8191";
       }
       description
         "Defines type for MEPIDm range is 1..8191";
     }

     typedef CCM-Interval {
       default "interval-1min";
       type enumeration {
         enum "interval-invalid" {
           value 0;
```

```
         }
         enum "interval-300hz" {
           value 1;
         }
         enum "interval-10ms" {
           value 2;
         }
         enum "interval-100ms" {
           value 3;
         }
         enum "interval-1s" {
           value 4;
         }
         enum "interval-10s" {
           value 5;
         }
         enum "interval-1min" {
           value 6;
         }
         enum "interval-10min" {
           value 7;
         }
       }
       reference
         "802.2Q Rev5 or 802.ag, all of the above
          are standard enumeration from the 802.1Q";
       description
         "IntervalInvalid - value 0
          Interval300Hz - Value 1
          Intervale10ms - value 2
          Interval100ms - value3
          Interval1s    - value 4
          Interval10s  - value 5
          Interval1min - value 6
          Interval10min - value 7";
     }

     typedef ecmp-choices {
       type enumeration {
         enum "ecmp-use-platform-hash" {
           value 0;
         }
         enum "ecmp-use-round-robin" {
           value 1;
         }
       }
     }
```

```
     typedef MD-name-format {
       type enumeration {
         enum "ieee-reserved" {
           value 0;
         }
         enum "none" {
           value 1;
         }
         enum "dns-like-name" {
           value 2;
         }
         enum "mac-address-and-uint" {
           value 3;
           reference "802.1Q Rev5";
           description
             "Domain name 3 specifies domain name is mac-address + 2
   octets.";
         }
       }
       reference "802.1Q";
       description
         "defines the domain name format";
     }

     typedef MA-name-format {
       type enumeration {
         enum "ieee-reserved" {
           value 0;
         }
         enum "primary-vid" {
           value 1;
         }
         enum "char-string" {
           value 2;
         }
         enum "unsigned-int16" {
           value 3;
         }
         enum "rfc2865-vpnid" {
           value 4;
         }
       }
       reference "802.1Q";
       description
         "Defines Format of MA-names";
     }
```

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

     grouping mep-address {
       choice mep-address {
         case mac-address {
           leaf mac-address {
             type yang:mac-address;
           }
         }
         case ipv4-address {
           leaf ipv4-address {
             type inet:ipv4-address;
           }
         }
         case ipv6-address {
           leaf ipv6-address {
             type inet:ipv6-address;
           }
         }
       }
     }

     grouping maintenance-domain {
       status current;
       description
         "Defines the MA-domain group";
       reference "802.1Q Rev5";
       leaf technology {
         mandatory true;
         status current;
         type identityref {
           base technology-types;
         }
         description
           "Defines the technology";
       }
       leaf md-name-format {
         mandatory true;
         status current;
         description
           "Defines the maintenance domain name";
         type MD-name-format;
```

```
        reference "802.1Q Rev5";
      }
      leaf md-name {
        status current;
        description
          "Defines the MA-Domain name. This is a binary (octet) string
           of 43 bytes";
        type binary {
          length "1..43";
        }
        reference "802.1Q Rev5";
      }
      leaf md-level {
        mandatory true;
        status current;
        description
          "Defines the MD-Level";
        type int32 {
          range "0..7";
        }
        reference "802.1Q Rev5 or 802.1ag";
      }
    }

    grouping ma-identifier {
      description
        "ma-identifier defines MAID parameters as defined in 8021Q";
      reference "IEEE 802.1Q Rev5";
      leaf ma-name-format {
        mandatory true;
        status current;
        description
          "This defines the MA name format 1 is no format,
           2 - dnslikename, 3- macaddress 4-CharString";
        type MA-name-format;
        reference "IEEE 802.1Q Rev 5";
      }
      leaf ma-name {
        mandatory true;
        description
          "Define the MA-Name according to the specified format.
           This is 43 byte string.";
        type binary {
          length "1..45";
        }
        reference "802.1Q Rve 5 or 8021ag Clause 21.6.5";
      }
```

```
     }

     grouping MEP {
       status current;
       description
         "Defines elements withing the MEP";
       reference "802.1Q Rev5";
       leaf mep-id {
         mandatory true;
         status current;
         description
           "Assigm MEPID in the range of 1..8191";
         type MEP-id {
           range "1..8191";
         }
         reference "802.1Q Rev5";
       }
       leaf mep-name {
         type string;
         description
           "Defines textual name for MEP. This is not specified in IEEE
   but
            defined in IETF OAM for ease of use";
       }
       leaf mep-direction {
         type MEP-direction;
         mandatory true;
       }
       leaf ccm-Tx-enable {
         type boolean;
         default "false";
       }
       uses mep-address;
       uses context-id;
       leaf Interface {
         type if:interface-ref;
         description
           "Interface name as defined by ietf-interfaces";
       }
     }

     grouping CCM-defect-stats {
       description
         "Contains all of  the CCM related defect stats";
       leaf ccm-rdi-indicator {
         config false;
         type boolean;
```

```
          description
            "True indicate one or more of the MEP have seen RDI
             flag set from remote MEP";
        }
        leaf ccm-xcon-count {
          config false;
          type oam-counter32;
          description
            "Number of times cross connect errors are seen";
        }
        leaf ccm-xcon-Indicator {
          config false;
          type boolean;
          description
            "There is currently cross connect error seen since last
             clearing of the variable";
        }
      }

      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 {
          units "milliseconds";
          type oam-counter32;
          description
            "Delay is specified in milliseconds";
        }
        leaf average-delay {
          units "milliseconds";
          type oam-counter32;
          description
            "average delay in milliseconds";
        }
        leaf max-delay {
          type oam-counter32;
          units "millisecond";
        }
      }
```

```
     grouping MIP {
       description
         "defines MIP";
       leaf interface {
         type if:interface-ref;
       }
       leaf direction {
         type MEP-direction;
       }
     }

     grouping nested-oam-layer {
       leaf offset {
         type int8 {
           range "1..7";
         }
         description
           "defines nested OAM layer offset
            +1 is the layer immediatly above
            -1 is the layer immediatly below";
       }
       uses maintenance-domain;
       uses ma-identifier;
     }

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

```
     grouping context-id {
       description
         "grouping for context id, this will be augmented
          by others who use this component";
       choice context-id {
         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";
           }
         }
       }
     }

     grouping flow-entropy {
       description
         "defines the grouping statement for flow-entropy";
       choice flow-entropy {
         case flow-entropy-null;
       }
     }

     container domains {
       status current;
       config true;
       description
         "Contains configuration related data. Within the container
          is list of fault domains. Wihin each domian has List of MA.";
       list domain {
         uses maintenance-domain {
           status current;
         }
         key "md-name technology";
         ordered-by system;
         status current;
         config true;
         description
           "Define the list of Domains within the IETF-OAM";
         container MAs {
           presence
             "Indicates creation of MA within the Domain
              There can be more than one MA within a specified domain";
           status current;
```

```
           config true;
           description
             "This container defines MA, within that have multiple MA
              and within MA have MEP, MIP";
           list MA {
             ordered-by system;
             status current;
             config true;
             key "ma-name";
             uses ma-identifier;
             uses context-id;
             leaf ccm-Interval {
               default "interval-invalid";
               description
                 "Defines CCM Interval 0- Means disable
                  1 - CCM are sent 3 1/3 ms
                  2 - CCM are sent every 10 ms
                  3- CCM are sent every 100 ms
                  4- CCM are sent every 1 s
                  5 - CCM are sent every 10 s
                  6 - CCM are sent every 1 minute
                  7- CCM are sent every 10 mins";
               type CCM-Interval;
               reference "802.1Q Rev5 and 802.1ag";
             }
             leaf ccm-loss-threshold {
               default "3";
               type uint32;
               description
                 "number of consecutive CCM messages missed before
                  declaring RDI fault. This is monitored per each
                  remote MEP";
             }
             leaf ccm-ttl {
               type uint8;
               default "255";
             }
             uses flow-entropy;
             list MEP {
               key "mep-id";
               ordered-by system;
               status current;
               config true;
               description
                 "contain list of MEPS";
               uses MEP {
                 status current;
```

```
               }
               uses interface-status {
                 description
                   "status of associated interface";
               }
               uses flow-entropy;
               list session {
                 key "user-cookie destination-mepid";
                 ordered-by user;
                 config true;
                 description
                   "per session basis create the monitoring";
                 leaf user-cookie {
                   config true;
                   type uint32;
                   description
                     "user need to specify some cookie to identify
                      multiple sessions between two MEPs";
                 }
                 leaf ttl {
                   config true;
                   type uint8;
                   default "255";
                 }
                 leaf interval {
                   units "milliseconds";
                   default "1000";
                   type uint32;
                   description
                     "In milli seconds. 0 means continous";
                 }
                 leaf enable {
                   default "false";
                   config true;
                   type boolean;
                   description
                     "enable or disable a monitor session";
                 }
                 leaf ecmp-choice {
                   config true;
                   type ecmp-choices;
                   description
                     "0 means use the specified interface
                      1 means use round robin";
                 }
                 leaf destination-mepid {
                   type MEP-id;
```

```
                  }
                  container destination-mep-address {
                    uses mep-address;
                  }
                  uses CCM-defect-stats;
                  uses context-id;
                  uses flow-entropy;
                  list outgoing-interface {
                    config true;
                    key "interface";
                    leaf interface {
                      type leafref {
                        path "/if:interfaces/if:interface/if:name";
                      }
                      config true;
                    }
                  }
                }
              }
              list remote-MEP {
                key "mep-id";
                ordered-by system;
                status current;
                config true;
                description
                  "list all of the remote MEP within the MA";
                leaf mep-id {
                  mandatory true;
                  status current;
                  description
                    "Assigm MEPID in the range of 1..8191";
                  config true;
                  type uint32;
                  reference "802.1Q Rev5";
                }
                uses mep-address;
                leaf mep-name {
                  type string;
                  description
                    "Defines textual name for MEP. This is not
                     specified in IEEE but defined in IETF OAM
                     for ease of use";
                }
                leaf ccm-rx-error-count {
                  type oam-counter32;
                  description
                    "counts number of CCM packets that was
```

```
                    expected but not received";
               }
             }
             list MIP {
               key "interface direction";
               uses MIP;
             }
             uses CCM-defect-stats {
               description
                 "CCM defect stats capture at MA level
                  This will contain aggregrate stats from all MEP";
             }
             list nested-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 effeciently troubleshoot a connectivity issue";
               uses nested-oam-layer;
             }
           }
         }
       }
     }
     notification CCM-RDI-notification {
       description
         "When RDI is received this notificiation is sent";
       leaf mep-id {
         type MEP-id;
         description
           "Indicate which MEP is seeing the error";
       }
       leaf remote-mepid {
         type MEP-id;
         description
           "Who is seeing the error (if known) if unknown make it 0.";
       }
       leaf error-message {
         type string {
           length "0..255";
         }
         description
           "Error message to indicate more details.";
       }
     }
     rpc ping {
```

```
       description
         "Generates Ping and return response";
       input {
         uses maintenance-domain {
           description
             "Specifies the MA-domain";
         }
         uses ma-identifier {
           description
             "identfies the Maintenance association";
         }
         uses context-id;
         uses flow-entropy;
         leaf ttl {
           type uint8;
           default "255";
         }
         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;
           }
         }
         container source-mep {
           uses mep-address;
           leaf mep-id {
             type MEP-id;
           }
         }
         container destination-mep {
           uses mep-address;
           leaf mep-id {
             type MEP-id;
           }
```

```
        }
      }
      output {
        uses monitor-stats {
          description
            "Stats of Ping is same as that of monitor sessions";
        }
      }
    }
    rpc trace-route {
      description
        "Generates Trace-route and return response. Starts with TTL
         of one and increment by one at each hop. Untill destination
         reached or TTL reach max valune";
      input {
        uses maintenance-domain {
          description
            "Specifies the MA-domain";
        }
        uses ma-identifier {
          description
            "identfies the Maintenance association";
        }
        uses context-id;
        uses flow-entropy;
        leaf ttl {
          type uint8;
          default "255";
        }
        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;
          }
```

```
         }
         container source-mep {
           uses mep-address;
           leaf mep-id {
             type MEP-id;
           }
         }
         container destination-mep {
           uses mep-address;
           leaf mep-id {
             type MEP-id;
           }
         }
       }
       output {
         list response {
           key "ttl";
           leaf ttl {
             type uint8;
           }
           container destination-mep {
             uses mep-address;
             leaf mep-id {
               type MEP-id;
             }
           }
           uses monitor-stats;
         }
       }
     }
   }

   <CODE ENDS>
```

Figure 6 YANG module of OAM

7. Base Mode for IP

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.

7.1. MEP Address

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

7.2. 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. 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 IP we propose to use MEP ID zero (0) as the default MEP-ID.

CCM packet use MEP-ID on the paylod. CCM MUST NOT be used in the Base Mode for IP. 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.

[8021Q] 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 of IP and MUST NOT be used for other purposes.

7.3. Maintenance Domain

Default MD-LEVEL is set to 3.

7.4. Maintenance Association

MAID [8021Q] 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 [8021Q]) and Short MA name set to 65532 (0xFFFC).

8. Security Considerations

TBD

9. IANA Considerations

This document registers the following namespace URI in the IETF XML registry.

URI:TBD

10. References

10.1. Normative References

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

[RFC792] Postel, J., "Internet Control Message Protocol", STD 5,RFC 792, September 1981.

[8021Q] IEEE, "Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks", IEEE Std 802.1Q-2011, August, 2011.

10.2. Informative References

[Y1731] ITU, "OAM functions and mechanisms for Ethernet based networks", ITU-T G.8013/Y.1731, July, 2011.

[RFC7174] Salam, S., et.al., "TRILL OAM Framework", RFC7174, May 2014.

[RFC6291] Andersson, L., et.al., "Guidelines for the use of the "OAM" Acronym in the IETF" RFC 6291, June 2011.

[RFC6325] Perlman, R., et.al., "Routing Bridges (RBridges): Base Protocol Specification", RFC 6325, July 2011.

[OAMOVW] Mizrahi, T., et.al., "An Overview of Operations, Administration, and Maintenance (OAM) Tools", draft-ietf-opsawg-oam-overview-16, Work in Progress, March 2014.

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

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

[RFC6371] Busi, I., et.al., "Operations, Administration, and Maintenance Framework for MPLS-Based Transport Networks, RFC 6317, September 2011.

[TRILLOAMFM] Senevirathne, T., et.al., "TRILL OAM Fault Management", draft-ietf-trill-oam-fm, Work in Progress, May 2014.

11. 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.

This document was prepared using 2-Word-v2.0.template.dot.

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.

Email: nfinn@cisco.com

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

Email: dekumar@cisco.com

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

Email: ssalam@cisco.com

Carlos Pignataro
Cisco Systems
7200-12 Kit Creek Rd,
Research Triangle Park, NC 27709

Email: cpignata@cisco.com