Internet Engineering Task Force (IETF)
Request for Comments: 8531
Category: Standards Track
ISSN: 2070-1721

D. Kumar
Cisco
Q. Wu
M. Wang
Huawei

April 2019

Generic YANG Data Model for Connection-Oriented Operations, Administration, and Maintenance (OAM) Protocols

Abstract

This document presents a base YANG data model for connection-oriented Operations, Administration, and Maintenance (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).

The YANG data model in this document conforms to the Network Management Datastore Architecture.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8531.

Table of Contents

1. Introduction

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

1. monitor network communications (i.e., reachability verification and Continuity Check)

2. troubleshoot failures (i.e., fault verification and localization)

3. monitor service-level agreements and performance (i.e., performance management)

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

The different sets of OAM tools may support both connection-oriented technologies or connectionless technologies. In connection-oriented technologies, a connection is established prior to the transmission of data. After the connection is established, no additional control information such as signaling or operations and maintenance information is required to transmit the actual user data. In connectionless technologies, data is typically sent between communicating endpoints without prior arrangement, but control information is required to identify the destination (e.g., [G.800]). The YANG data model for OAM protocols using connectionless communications is specified in [RFC8532] and [IEEE802.1Q].

Connectivity Fault Management as specified in [IEEE802.1Q] is a well-established OAM standard that is widely adopted for Ethernet networks. ITU-T [G.8013], MEF Forum (MEF) Service OAM [MEF-17], MPLS Transport Profile (MPLS-TP) [RFC6371], and Transparent Interconnection of Lots of Links (TRILL) [RFC7455] all define OAM mechanisms based on the manageability framework of Connectivity Fault Management (CFM) [IEEE802.1Q].

Given the wide adoption of the underlying OAM concepts defined in CFM [IEEE802.1Q], 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 CFM [IEEE802.1Q] model and extend it to a technology-independent framework and define the corresponding YANG data model accordingly. The YANG data model presented in this document is the base model for connection-oriented OAM protocols and supports generic continuity check, connectivity verification, and path discovery (traceroute). The generic YANG data model for connection-oriented OAM is designed to be extensible to other connection-oriented technologies. Technology-dependent nodes

and remote procedure call (RPC) commands are defined in technology-specific YANG data models, which use and extend the base model defined here. As an example, Virtual eXtensible Local Area Network (VXLAN) uses the source UDP port number for flow entropy, while TRILL uses either (a) MAC addresses, (b) the VLAN tag or Fine-Grained Label, and/or (c) IP addresses for flow entropy in the hashing for multipath selection. To capture this variation, corresponding YANG data models would define the applicable structures as augmentation to the generic base model presented here. This accomplishes three goals: First, it keeps each YANG data model smaller and more manageable. Second, it allows independent development of corresponding YANG data models. Third, implementations can limit support to only the applicable set of YANG data models (e.g., TRILL RBridge may only need to implement the generic model and the TRILL YANG data model).

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 issue 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 connectivity from device A loopback 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 CFM [IEEE802.1Q]. A user, upon detecting the loopback failure, may decide to drill down to the lower level at different segments of the path and issue the corresponding fault verification (Loopback Message) and fault isolation (Looktrace Message) 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 data model presented in this document facilitates that without needing changes to the underlying protocols.

The YANG data model in this document conforms to the Network Management Datastore Architecture defined in [RFC8342].

2. Conventions Used in This Document

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

Many of the terms used in this document (including those set out in Sections Section 2.1 and Section 2.2) are specific to the world of OAM. This document does not attempt to explain the terms but does assume that the reader is familiar with the concepts. For a good overview, read [IEEE802.1Q]. For an example of how these OAM terms appear in IETF work, see [RFC6371].

2.1. Abbreviations

CCM - Continuity Check Message [IEEE802.1Q]

ECMP - Equal-Cost Multipath

LBM - Loopback Message [IEEE802.1Q]

LTM - Linktrace Message [IEEE802.1Q]

MP - Maintenance Point [IEEE802.1Q]

MEP - Maintenance End Point [RFC7174] (also known as Maintenance association End Point [IEEE802.1Q] and MEG End Point [RFC6371])

MIP - Maintenance Intermediate Point [RFC7174] (also known as Maintenance domain Intermediate Point [IEEE802.1Q] and 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 Procedure Call

CC - Continuity Check [RFC7276]

CV - Connectivity Verification [RFC7276]

2.2. Terminology

Continuity Checks - Continuity Checks are used to verify that a destination is reachable and therefore also are referred to as "reachability verification".

Connectivity Verification - Connectivity Verification is used to verify that a destination is connected. It is 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.

Proactive OAM - Proactive OAM refers to OAM actions that are carried out continuously to permit proactive reporting of fault. A proactive OAM method requires persistent configuration.

On-demand OAM - On-demand OAM refers to OAM actions that are initiated via manual intervention for a limited time to carry out diagnostics. An on-demand OAM method requires only transient configuration.

2.3. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

3. Architecture of Generic YANG Data Model for Connection-Oriented OAM

In this document, we define a generic YANG data model for connection-oriented OAM protocols. The YANG data model defined here is generic in a sense that other technologies can extend it for technology-specific needs. The generic YANG data model for connection-oriented OAM acts as the root for other OAM YANG data 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 data models to the Generic YANG Data Model for connection-oriented OAM. The Generic YANG data model for connection-oriented OAM provides a framework where technology-specific YANG data models can inherit constructs from the base YANG data models without needing to redefine them within the sub-technology.

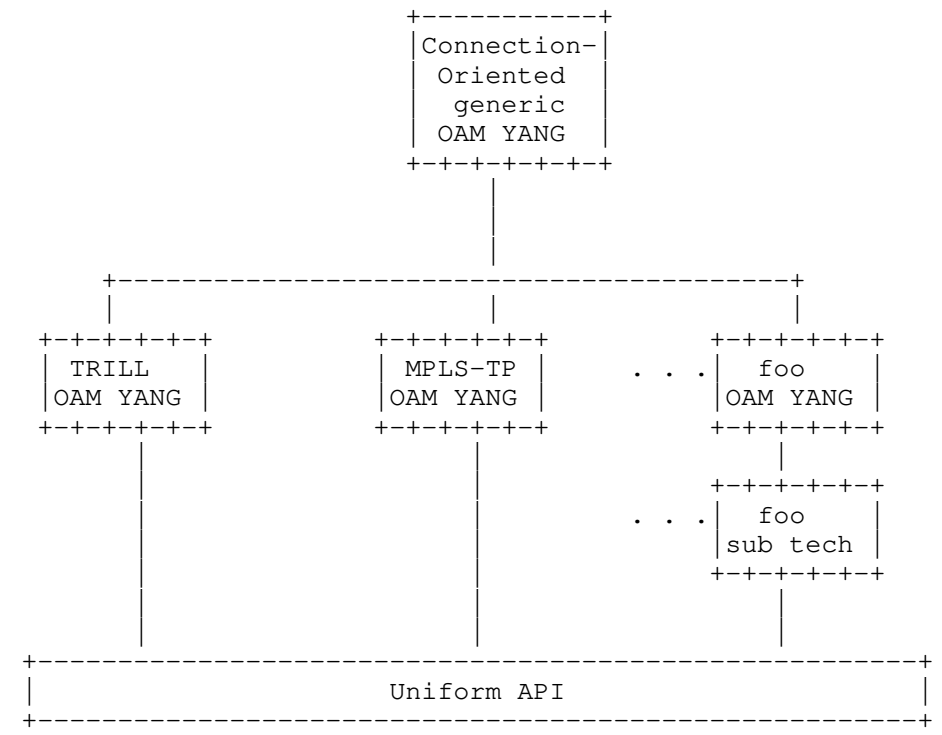

Figure 1: Relationship of OAM YANG Data Model to Generic (Base) YANG Data Model

4. Overview of the Connection-Oriented OAM YANG Data Model

In this document, we adopt the concepts of the CFM [IEEE802.1Q] model and structure 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 Associations (MAs). In TRILL, the MA can correspond to a Fine-Grained Label.

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 data model presented here provides flexibility to accommodate different addressing schemes.

Commands are presented in the management protocol, which is orthogonal to the Maintenance Domain. These are RPC commands, in YANG terms. They provide uniform APIs for Continuity Check, connectivity verification, path discovery (traceroute), and their equivalents, as well as other OAM commands.

The OAM entities in the generic YANG data model defined here will be either explicitly or implicitly configured using any of the OAM tools. The OAM tools used here are limited to the OAM toolset specified in Section 5.1 of [RFC7276]. In order to facilitate a 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 model's 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 data model to support Base Mode. Implementations that comply with this document use, by default, 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", a separate list is maintained per MD. The MD list uses the key "md-name-string" for indexing. The "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. The "md-name-format" indicates the format of the augmented "md-name". The "md-name" is presented as choice/case construct. Thus, it is easily augmentable by derivative work.

```
module: ietf-connection-oriented-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 (MAs). 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-connection-oriented-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 (MAs)

4.3. Maintenance End Point (MEP) Configuration

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

```
module: ietf-connection-oriented-oam
 +--rw domains
    +--rw domain* [technology md-name-string]
       +--rw technology                  identityref
       .
       .
       +--rw mas
          +--rw ma* [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
             |  |  +--:(ip-address)
             |  |     +--rw ip-address?      inet:ip-address
               .          .
               .          .
               .          .
```

Snippet of Data Hierarchy Related to Maintenance End Point (MEP)

4.4. RPC Definitions

The RPC model facilitates issuing commands to a "server" (in this case, to the device that need to execute the OAM command) and obtaining a response. The 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 4.5 for the complete data hierarchy and Section 5 for the YANG module.

```
module: ietf-connection-oriented-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 md-level?      -> /domains/domain/md-level
  |  |  +---w ma-name-string -> /domains/domain/mas/ma/ma-name-string
  |  |  +---w cos-id?                 uint8
  |  |  +---w 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
  |  |  |  |  +--:(ip-address)
  |  |  |  |     +---w ip-address?      inet:ip-address
  |  |  |  +---w (mep-id)?
  |  |  |  |  +--:(mep-id-int)
  |  |  |  |     +---w mep-id-int?      int32
  |  |  |  +---w mep-id-format?   identityref
  |  |  +---w count?                  uint32
  |  |  +---w cc-transmit-interval?   time-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 md-level?      -> /domains/domain/md-level
  |  |  +---w ma-name-string -> /domains/domain/mas/ma/ma-name-string
  |  |  +---w cos-id?            uint8
  |  |  +---w 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
  |  |  |  |  +--:(ip-address)
```

```
      |  |  |  |     +---w ip-address?      inet:ip-address
      |  |  |  +---w (mep-id)?
      |  |  |  |  +--:(mep-id-int)
      |  |  |  |     +---w mep-id-int?      int32
      |  |  |  +---w mep-id-format?   identityref
      |  |  +---w count?             uint32
      |  |  +---w interval?          time-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 md-level?      -> /domains/domain/md-level
         |  +---w ma-name-string -> /domains/domain/mas/ma/ma-name-string
         |  +---w cos-id?             uint8
         |  +---w 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
         |  |  |  +--:(ip-address)
         |  |  |     +---w ip-address?      inet:ip-address
         |  |  +---w (mep-id)?
         |  |  |  +--:(mep-id-int)
         |  |  |     +---w mep-id-int?      int32
         |  |  +---w mep-id-format?   identityref
         |  +---w count?              uint32
         |  +---w interval?           time-interval
         +--ro output
            +--ro response* [response-index]
               +--ro response-index     uint8
               +--ro ttl?               uint8
               +--ro destination-mep
               |  +--ro (mep-address)?
               |  |  +--:(mac-address)
               |  |  |  +--ro mac-address?     yang:mac-address
               |  |  +--:(ip-address)
               |  |     +--ro ip-address?      inet:ip-address
               |  +--ro (mep-id)?
               |  |  +--:(mep-id-int)
               |  |     +--ro mep-id-int?      int32
               |  +--ro mep-id-format?   identityref
               +--ro mip {mip}?
```

```
               |  +--ro interface?     if:interface-ref
               |  +--ro (mip-address)?
               |     +--:(mac-address)
               |     |  +--ro mac-address?   yang:mac-address
               |     +--:(ip-address)
               |        +--ro ip-address?    inet:ip-address
               +--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 upon detecting a defect condition and upon clearing a defect with a 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 technology-specific YANG modules that augment the generic YANG data model to provide statistics due to proactive OAM-like Continuity Check Messages -- for example, CCM transmit, CCM receive, CCM error, etc.

4.7. OAM Data Hierarchy

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

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

```
                +--rw (connectivity-context)?
                |  +--:(context-null)
                |     +--rw context-null?     empty
                +--rw 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
                |  |  +--:(ip-address)
                |  |     +--rw ip-address?      inet:ip-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
                |     |     +--:(ip-address)
                |     |        +--rw ip-address?    inet:ip-address
                |     +--rw cos-id?                   uint8
                +--rw mip* [name] {mip}?
                   +--rw name           string
                   +--rw interface?     if:interface-ref
                   +--rw (mip-address)?
                      +--:(mac-address)
                      |  +--rw mac-address?   yang:mac-address
                      +--:(ip-address)
                         +--rw ip-address?    inet:ip-address

   rpcs:
     +---x continuity-check {continuity-check}?
     |  +---w input
     |  |  +---w technology?             identityref
     |  |  +---w md-name-string -> /domains/domain/md-name-string
     |  |  +---w md-level?      -> /domains/domain/md-level
     |  |  +---w ma-name-string -> /domains/domain/mas/ma/ma-name-string
```

```
      |   |  +---w cos-id?                 uint8
      |   |  +---w 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
      |   |  |  |  +--:(ip-address)
      |   |  |  |     +---w ip-address?      inet:ip-address
      |   |  |  +---w (mep-id)?
      |   |  |  |  +--:(mep-id-int)
      |   |  |  |     +---w mep-id-int?      int32
      |   |  |  +---w mep-id-format?   identityref
      |   |  +---w count?                  uint32
      |   |  +---w cc-transmit-interval?   time-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 md-level?      -> /domains/domain/md-level
      |  |  +---w ma-name-string -> /domains/domain/mas/ma/ma-name-string
      |  |  +---w cos-id?            uint8
      |  |  +---w 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
      |  |  |  |  +--:(ip-address)
      |  |  |  |     +---w ip-address?      inet:ip-address
      |  |  |  +---w (mep-id)?
      |  |  |  |  +--:(mep-id-int)
      |  |  |  |     +---w mep-id-int?      int32
      |  |  |  +---w mep-id-format?   identityref
      |  |  +---w count?             uint32
      |  |  +---w interval?          time-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 md-level?      -> /domains/domain/md-level
      |  +---w ma-name-string -> /domains/domain/mas/ma/ma-name-string
      |  +---w cos-id?             uint8
      |  +---w 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
      |  |  |  +--:(ip-address)
      |  |  |     +---w ip-address?      inet:ip-address
      |  |  +---w (mep-id)?
      |  |  |  +--:(mep-id-int)
      |  |  |     +---w mep-id-int?      int32
      |  |  +---w mep-id-format?   identityref
      |  +---w count?              uint32
      |  +---w interval?           time-interval
      +--ro output
         +--ro response* [response-index]
            +--ro response-index     uint8
            +--ro ttl?               uint8
            +--ro destination-mep
            |  +--ro (mep-address)?
            |  |  +--:(mac-address)
            |  |  |  +--ro mac-address?     yang:mac-address
            |  |  +--:(ip-address)
            |  |     +--ro ip-address?      inet:ip-address
            |  +--ro (mep-id)?
            |  |  +--:(mep-id-int)
            |  |     +--ro mep-id-int?      int32
            |  +--ro mep-id-format?   identityref
            +--ro mip {mip}?
            |  +--ro interface?     if:interface-ref
            |  +--ro (mip-address)?
            |     +--:(mac-address)
            |     |  +--ro mac-address?   yang:mac-address
            |     +--:(ip-address)
            |        +--ro ip-address?    inet:ip-address
            +--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

This module imports typedefs from [RFC6991] and [RFC8343], and it references [RFC6371], [RFC6905], and [RFC7276].

<CODE BEGINS> file "ietf-connection-oriented-oam@2019-04-16.yang"

```
module ietf-connection-oriented-oam {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-connection-oriented-oam";
  prefix co-oam;

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

  organization
    "IETF LIME Working Group";
  contact
    "WG Web:    http://datatracker.ietf.org/wg/lime
     WG List:   <mailto:lime@ietf.org>
     Editor:    Deepak Kumar <dekumar@cisco.com>
     Editor:    Qin Wu <bill.wu@huawei.com>
     Editor:    Michael 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-independent manner.
     Functional-level abstraction is independent
     with YANG modeling. It is assumed that each protocol
     maps corresponding abstracts to its native format.
     Each protocol may extend the YANG data model defined
     here to include protocol-specific extensions

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

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

```
     (http://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC 8531; see
     the RFC itself for full legal notices.";

  revision 2019-04-16 {
    description
      "Initial revision.";
    reference
      "RFC 8531: Generic YANG Data Model for Connection-
       Oriented Operations, Administration, and Maintenance (OAM)
       Protocols";
  }

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

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

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

  feature mip {
    description
      "This feature indicates that the Maintenance
       Intermediate Point (MIP) needs to be explicitly configured";
```

```
  }

  identity technology-types {
    description
      "This is the base identity of technology types that are
       TRILL, MPLS-TP, etc.";
  }

  identity command-sub-type {
    description
      "Defines different RPC command subtypes,
       e.g., TRILL OAM as specified in RFC 6905; this is
       optional for most cases.";
    reference
      "RFC 6905: Requirements for OAM in Transparent
       Interconnection of Lots of Links (TRILL)";
  }

  identity on-demand {
    base command-sub-type;
    description
      "On-demand activation indicates that the tool is activated
       manually to detect a specific anomaly.
       An on-demand OAM method requires only transient configuration.";
    reference
      "RFC 7276: An Overview of Operations, Administration, and
       Maintenance (OAM) Tools";
  }

  identity proactive {
    base command-sub-type;
    description
      "Proactive activation indicates that the tool is activated on a
       continual basis, where messages are sent periodically, and errors
       are detected when a certain number of expected messages are not
       received.  A proactive OAM method requires persistent
       configuration.";
    reference
      "RFC 7276: An Overview of Operations, Administration, and
       Maintenance (OAM) Tools";
  }

  identity name-format {
    description
      "This defines the name format, CFM (IEEE 802.1Q) defines varying
       styles of names.  It is expected that name format is an identity
       reference 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 Defect Indication (RDI), loss of continuity.";
  }

  identity rdi {
    base defect-types;
    description
      "The RDI indicates the
       aggregate health of the remote Maintenance End Points (MEPs).";
  }

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

  identity loss-of-continuity {
    base defect-types;
    description
      "Indicates that there are no proactive Continuity Check (CC)
       OAM packets from the source MEP (and in the case of
       Connectivity Verification, this includes the requirement to have
       the expected unique, technology-dependent source MEP identifier)
       received within the interval.";
    reference
      "RFC 6371: Operations, Administration, and Maintenance
```

```
       Framework for MPLS-Based Transport Networks";
  }

  identity cv-defect {
    base defect-types;
    description
      "This function should support monitoring between the MEPs
       and, in addition, between a MEP and MIP.  When performing
       Connectivity Verification, the Continuity Check and
       Connectivity Verification (CC-V) messages need to include
       unique identification of the MEG that is being monitored and
       the MEP that originated the message.";
    reference
      "RFC 6371: Operations, Administration, and Maintenance
       Framework for MPLS-Based Transport Networks";
  }

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

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

  typedef time-interval {
    type decimal64 {
      fraction-digits 2;
    }
    units "milliseconds";
    description
      "Time interval between packets in milliseconds.
       Time interval should not be less than 0.
```

```
       0 means no packets are sent.";
  }

  typedef md-name-string {
    type string;
    description
      "Generic administrative name for Maintenance Domain (MD).";
  }

  typedef ma-name-string {
    type string;
    description
      "Generic administrative name for a
       Maintenance Association (MA).";
  }

  typedef oam-counter32 {
    type yang:zero-based-counter32;
    description
      "Define 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 (e.g., protocol in layer 0 to layer 7).";
  }

  grouping maintenance-domain-reference {
    description
      "This grouping uniquely identifies a Maintenance Domain.";
    leaf maintenance-domain {
      type leafref {
        path "/co-oam:domains/co-oam:domain/co-oam:md-name-string";
      }
      description
        "A reference to a specific Maintenance Domain.";
    }
  }

  grouping maintenance-association-reference {
    description
      "This grouping uniquely identifies a
       Maintenance Association.  It consists
       of a maintenance-domain-reference and
```

```
       a maintenance-association leafref.";
    uses maintenance-domain-reference;
    leaf maintenance-association {
      type leafref {
        path "/co-oam:domains/co-oam:domain[co-oam:md-name-string "
           + "= current()/../maintenance-domain]/co-oam:mas"
           + "/co-oam:ma/co-oam:ma-name-string";
      }
      description
        "A reference to a specific Maintenance Association.";
    }
  }

  grouping maintenance-association-end-point-reference {
    description
      "This grouping uniquely identifies
       a Maintenance Association.  It consists
       of a maintenance-association-reference and
       a maintenance-association-end-point leafref.";
    uses maintenance-association-reference;
    leaf maintenance-association-end-point {
      type leafref {
        path "/co-oam:domains/co-oam:domain[co-oam:md-name-string "
           + "= current()/../maintenance-domain]/co-oam:mas"
           + "/co-oam:ma[co-oam:ma-name-string = "
           + "current()/../maintenance-association]"
           + "/co-oam:mep/co-oam:mep-name";
      }
      description
        "A reference to a specific Maintenance
         association End Point.";
    }
  }

  grouping time-to-live {
    leaf ttl {
      type uint8;
      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 to be defined; it will be defined in
             a 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 a technology.";
        }
      }
      description
        "Defect Message choices.";
    }
    description
      "Defect Message.";
  }

  grouping mep-address {
    choice mep-address {
      default "ip-address";
      case mac-address {
        leaf mac-address {
          type yang:mac-address;
          description
            "MAC Address.";
        }
        description
          "MAC Address based MEP Addressing.";
      }
      case ip-address {
        leaf ip-address {
          type inet:ip-address;
          description
            "IP Address.";
        }
        description
          "IP Address based MEP Addressing.";
      }
      description
        "MEP Addressing.";
    }
```

```
    description
      "Grouping for MEP Address";
  }

  grouping mip-address {
    choice mip-address {
      default "ip-address";
      case mac-address {
        leaf mac-address {
          type yang:mac-address;
          description
            "MAC Address of Maintenance Intermediate Point";
        }
        description
          "MAC Address based MIP Addressing.";
      }
      case ip-address {
        leaf ip-address {
          type inet:ip-address;
          description
            "IP Address.";
        }
        description
          "IP Address based MIP Addressing.";
      }
      description
        "MIP Addressing.";
    }
    description
      "MIP Address.";
  }

  grouping maintenance-domain-id {
    description
      "Grouping containing leaves sufficient to identify
       a Maintenance Domain.";
    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
        "Maintenance Domain Name format.";
    }
    choice md-name {
      case md-name-null {
        leaf md-name-null {
          when "derived-from-or-self(../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 "derived-from-or-self(../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 placeholder when
           no monitoring statistics are needed.";
        leaf monitor-null {
          type empty;
          description
            "There are no monitoring statistics to be defined.";
        }
      }
      description
        "Define the monitor stats.";
    }
  }

  grouping connectivity-context {
    description
      "Grouping defining the connectivity context for an MA,
       for example, an LSP for MPLS-TP.  This will be
       augmented by each protocol that uses this component.";
    choice connectivity-context {
      default "context-null";
      case context-null {
        description
          "This is a placeholder when no context is needed.";
        leaf context-null {
          type empty;
          description
            "There is no context to be defined.";
```

```
        }
      }
      description
        "Connectivity context.";
    }
  }

  grouping cos {
    description
      "Grouping for Priority used in transmitted packets,
       for example, in the CoS field in MPLS-TP.";
    leaf cos-id {
      type uint8;
      description
        "Class of Service (CoS) ID; this value is used to indicate
         Class of Service information .";
    }
  }

  grouping mip-grouping {
    uses mip-address;
    description
      "Grouping for MIP
       configuration.";
  }

  container domains {
    description
      "Contains configuration related data.  Within the
       container, there is a list of fault domains.  Each
       domain has a list of MAs.";
    list domain {
      key "technology md-name-string";
      description
        "Define a list of Domains within the
         ietf-connection-oriented-oam module.";
      uses maintenance-domain-id;
      uses md-name;
      leaf md-level {
        type md-level;
        description
          "Define the MD level.";
      }
      container mas {
        description
          "Contains configuration-related data.  Within the
           container, there is a list of MAs.  Each MA has a
           list of MEPs.";
```

```
        list ma {
          key "ma-name-string";
          uses ma-identifier;
          uses ma-name;
          uses connectivity-context;
          uses cos {
            description
              "Default class of service for this MA;
               it may be overridden for particular MEPs,
               sessions, or operations.";
          }
          leaf cc-enable {
            type boolean;
            description
              "Indicate whether the CC is enabled.";
          }
          list mep {
            key "mep-name";
            description
              "Contain a list of MEPs.";
            uses mep;
            uses cos;
            leaf cc-enable {
              type boolean;
              description
                "Indicate whether the CC is enabled.";
            }
            list session {
              key "session-cookie";
              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 {
          if-feature "mip";
          key "name";
          leaf name {
            type string;
            description
              "Identifier of Maintenance Intermediate Point";
          }
          leaf interface {
            type if:interface-ref;
            description
              "Interface.";
          }
          uses mip-grouping;
          description
            "List for MIP.";
        }
        description
          "Maintenance Association list.";
      }
    }
  }
}

notification defect-condition-notification {
  description
    "When the 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 the defect belongs to.";
    }
    leaf ma-name-string {
      type leafref {
        path "/domains/domain/mas/ma/ma-name-string";
      }
      mandatory true;
      description
        "Indicate which MA the defect is associated with.";
    }
    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
        "Indicate who is generating the defect (if known). If
         unknown, set it to 0.";
    }
    uses defect-message {
      description
        "Defect message to provide more details.";
    }
  }

  notification defect-cleared-notification {
    description
      "When the defect is cleared, 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 the defect belongs to";
    }
    leaf ma-name-string {
      type leafref {
        path "/domains/domain/mas/ma/ma-name-string";
      }
      mandatory true;
      description
        "Indicate which MA the defect is associated with.";
    }
    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
        "Indicate who is generating the defect (if known). If
         unknown, set it to 0.";
    }
    uses defect-message {
      description
        "Defect message to provide more details.";
    }
  }

  rpc continuity-check {
    if-feature "continuity-check";
    description
      "Generates Continuity Check as per Table 4 of RFC 7276.";
    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 the defect belongs to.";
      }
      leaf md-level {
        type leafref {
          path "/domains/domain/md-level";
        }
        description
          "The Maintenance Domain Level.";
      }
      leaf ma-name-string {
        type leafref {
          path "/domains/domain/mas/ma/ma-name-string";
        }
        mandatory true;
        description
          "Indicate which MA the defect is associated with.";
      }
      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 messages to be sent.";
      }
      leaf cc-transmit-interval {
        type time-interval;
        description
          "Time interval between echo requests.";
      }
      leaf packet-size {
        type uint32 {
          range "64..10000";
        }
        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 Connectivity Verification as per Table 4 in RFC 7276.";
    input {
      leaf md-name-string {
        type leafref {
          path "/domains/domain/md-name-string";
        }
        mandatory true;
        description
          "Indicate which MD the defect belongs to.";
      }
      leaf md-level {
        type leafref {
          path "/domains/domain/md-level";
```

```
        }
        description
          "The Maintenance Domain Level.";
      }
      leaf ma-name-string {
        type leafref {
          path "/domains/domain/mas/ma/ma-name-string";
        }
        mandatory true;
        description
          "Indicate which MA the defect is associated with.";
      }
      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 messages to be sent.";
      }
      leaf interval {
        type time-interval;
        description
          "Time interval between echo requests.";
      }
```

```
      leaf packet-size {
        type uint32 {
          range "64..10000";
        }
        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 returns response.
       References RFC 7276 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 increments
       by one at each hop until the destination is reached or TTL
       reaches max value.";
    input {
      leaf md-name-string {
        type leafref {
          path "/domains/domain/md-name-string";
        }
        mandatory true;
        description
          "Indicate which MD the defect belongs to.";
      }
      leaf md-level {
        type leafref {
          path "/domains/domain/md-level";
        }
        description
          "The Maintenance Domain Level.";
      }
      leaf ma-name-string {
        type leafref {
          path "/domains/domain/mas/ma/ma-name-string";
        }
        mandatory true;
        description
          "Indicate which MA the defect is associated with.";
```

```
      }
      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 be sent at each TTL.";
      }
      leaf interval {
        type time-interval;
        description
          "Time 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.";
          }
        }
        container mip {
          if-feature "mip";
          leaf interface {
            type if:interface-ref;
            description
              "MIP interface.";
          }
          uses mip-address;
          description
            "MIP responding with traceroute";
        }
        uses monitor-stats {
          description
            "Stats of traceroute.";
        }
        description
          "List of responses.";
      }
    }
  }
}
```

<CODE ENDS>

6. Base Mode

The Base Mode ("default mode" described in Section 4) defines the default configuration that MUST be present in the devices that comply with this document. Base Mode allows users to have a "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 MEP associated with a virtual OAM port with no physical layer (NULL PHY). The MEP-ID associated with this MEP is zero (0). The choice of MEP-ID of zero is explained below.

MEP-ID is a 2-octet field by default. It is never used on the wire except when using CCM. It is important to have a method that can derive the MEP-ID of Base Mode in an automatic manner with no user intervention. The IP address cannot be directly used for this purpose, as the MEP-ID is a much smaller field. For the Base Mode of operation, MEP-ID is set to zero by default.

The CCM packet uses the MEP-ID in 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 values for the corresponding MEP IDs.

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

6.3. Maintenance Association

The ID of the Maintenance Association (MA-ID) [IEEE802.1Q] has a flexible format and includes two parts: Maintenance Domain Name and Short MA name. In the Base Mode of operation, the value of the Maintenance Domain Name must be the character string "GenericBaseMode" (excluding the quotes). In the Base Mode

operation, the Short MA Name format is set to a 2-octet integer format (value 3 in Short MA Format field [IEEE802.1Q]) and the Short MA name is set to 65532 (0xFFFC).

7. Connection-Oriented OAM YANG Data Model Applicability

The "ietf-connection-oriented-oam" module defined in this document provides a technology-independent abstraction of key OAM constructs for connection-oriented protocols. This module 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 Data Model Extension for TRILL OAM

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

In addition,the TRILL OAM YANG module also requires the base TRILL module ([TRILL-YANG]) to be supported, as there is a strong relationship between those modules.

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

7.1.1. MD Configuration Extension

MD level configuration parameters are management information that can be inherited in the TRILL OAM model and set by the 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 (i.e., at root level), the domain data node can be augmented with technology type.

Note that MD level configuration parameters provide context information for the management system to correlate faults, defects, and network failures with location information; this 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 co-oam:technology-types;
 description
  "trill type";
}
```

7.1.2. MA Configuration Extension

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

Note that MA level configuration parameters provide context information for the management system to correlate faults, defects, and network failures with location information; this 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-Grained 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 a VLAN ID or Fine-Grained Label.

```
   augment /co-oam:domains/co-oam:domain
/co-oam:mas/co-oam:ma/co-oam: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 a 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 the base model to add a nickname case to the MEP address choice node as follows:

```
augment /co-oam:domains/co-oam:domain
/co-oam:mas/co-oam:ma/co-oam:mep/co-oam:mep-address:
         +--:( mep-address-trill)
         |  +--rw mep-address-trill?  tril-rb-nickname
```

In addition, at the Maintenance association End Point (MEP) level (i.e., at the third level), the MEP data node can be augmented with an 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 Layer-Independent OAM Management in the Multi-Layer Environment (LIME) model extension for TRILL OAM. The snippet below illustrates its extension.

```
   augment /co-oam:domains/co-oam:domain
/co-oam:mas/co-oam:ma/co-oam:mep:
            +--rw flow-entropy-trill?   flow-entropy-trill
   augment /co-oam:domains/co-oam:domain
/co-oam:mas/co-oam:ma/co-oam:mep/co-oam:session:
            +--rw flow-entropy-trill?   flow-entropy-trill
```

7.1.4. RPC Extension

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

```
augment /co-oam:continuity-check/co-oam: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 /co-oam:continuity-check/co-oam:input:
         +--ro flow-entropy-trill?   flow-entropy-trill
augment /co-oam:continuity-check/co-oam:output:
      +--ro upstream-rbridge?   tril-rb-nickname
      +--ro next-hop-rbridge*   tril-rb-nickname
augment /co-oam:path-discovery/co-oam: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 /co-oam:path-discovery/co-oam:input:
         +--ro flow-entropy-trill?   flow-entropy-trill
augment /co-oam:path-discovery/co-oam:output/co-oam:response:
      +--ro upstream-rbridge?   tril-rb-nickname
      +--ro next-hop-rbridge*   tril-rb-nickname
```

7.2. Generic YANG Data Model Extension for MPLS-TP OAM

The MPLS-TP OAM YANG module can augment the connection-oriented OAM module with some technology-specific details. [MPLS-TP-OAM-YANG] presents the YANG data model for MPLS-TP OAM.

The configuration extensions for connection-oriented OAM include the MD configuration extension, Technology type extension, Technology Subtype extension, MA configuration extension, and MEP Configuration extension.

7.2.1. MD Configuration Extension

MD level configuration parameters are management information that can be inherited in the MPLS-TP OAM model and set by the connection-oriented OAM 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 (i.e., at root level), the domain data node can be augmented with technology type and technology subtype.

Note that MD level configuration parameters provide context information for the management system to correlate faults, defects, and network failures with location information; this 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-TP 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 co-oam:technology-types;
      description
       "mpls-tp type";
     }
```

7.2.1.2. Technology Subtype Extension

In MPLS-TP, since different encapsulation types such as IP/UDP encapsulation and PW-ACH encapsulation can be employed, the "technology-sub-type" data node is defined and added into the MPLS-TP OAM model to further identify the encapsulation types within the MPLS-TP OAM model. Based on it, we also define a technology subtype 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 subtype to
          further identify different encapsulations.
          Technology subtype is associated at the MA level."; }

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

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

         augment "/co-oam:domains/co-oam:domain"
               + "/co-oam:mas/co-oam:ma" {
                leaf technology-sub-type {
                  type identityref {
                    base technology-sub-type;
                  }
                }
              }
```

7.2.2. MA Configuration Extension

MA level configuration parameters are management information that can be inherited in the MPLS-TP OAM model and set by the connection-oriented OAM base model as default values. Examples of MA Name are MPLS-TP LSP MEG_ID, MEG Section ID, or MEG PW ID [RFC6370].

Note that MA level configuration parameters provide context information for the management system to correlate faults, defects, and network failures with location information; this 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 that can support an int32 value, and the same definition can be used for MPLS-TP with no further modification. In addition, at the Maintenance association End Point (MEP) level (i.e., at the third level), the MEP data node can be augmented with a session extension and interface extension.

8. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

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

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

/co-oam:domains/co-oam:domain/

/co-oam:domains/co-oam:domain/co-oam:mas/co-oam:ma

/co-oam:domains/co-oam:domain/co-oam:mas/co-oam:ma/co-oam:mep

/co-oam:domains/co-oam:domain/co-oam:mas/co-oam:ma/co-oam:mep/
co-oam: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]. The following registration has been made:

URI: urn:ietf:params:xml:ns:yang:ietf-connection-oriented-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-connection-oriented-oam
namespace: urn:ietf:params:xml:ns:yang:ietf-connection-oriented-oam
prefix: co-oam
reference: RFC 8531

10. References

10.1. Normative References

[IEEE802.1Q] IEEE, "IEEE Standard for Local and Metropolitan Area Networks-Bridges and Bridged Networks", IEEE Std 802.1Q.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

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

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://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, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

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

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

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

10.2. Informative References

[G.800] "Unified functional architecture of transport networks", ITU-T Recommendation G.800, 2016.

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

[MEF-17] MEF Forum, "Service OAM Requirements & Framework - Phase 1", MEF 17, April 2007.

[MPLS-TP-OAM-YANG] Zhang, L., Zheng, L., Aldrin, S., and G. Mirsky, "YANG Data Model for MPLS-TP Operations, Administration, and Maintenance (OAM)", Work in Progress, draft-zhang-mpls-tp-yang-oam-05, October 2017.

[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, <https://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, <https://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, <https://www.rfc-editor.org/info/rfc6371>.

[RFC6905] Senevirathne, T., Bond, D., Aldrin, S., Li, Y., and R. Watve, "Requirements for Operations, Administration, and Maintenance (OAM) in Transparent Interconnection of Lots of Links (TRILL)", RFC 6905, DOI 10.17487/RFC6905, March 2013, <https://www.rfc-editor.org/info/rfc6905>.

[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, <https://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, <https://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, <https://www.rfc-editor.org/info/rfc7455>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

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

[RFC8532] Kumar, D., Wang, M., Wu, Q., Ed., Rahman, R., and S. Raghavan, "Generic YANG Data Model for the Management of Operations, Administration, and Maintenance (OAM) Protocols That Use Connectionless Communications", RFC 8532, DOI 10.17487/RFC8532, April 2019, <https://www.rfc-editor.org/info/rfc8532>.

[TRILL-YANG] Weiguo, H., Yizhou, L., Kumar, D., Durrani, M., Zhai, H., and L. Xia, "TRILL YANG Data Model", Work in Progress, draft-ietf-trill-yang-04, December 2015.

[TRILL-YANG-OAM] Kumar, D., Senevirathne, T., Finn, N., Salam, S., Xia, L., and H. Weiguo, "YANG Data Model for TRILL Operations, Administration, and Maintenance (OAM)", Work in Progress, draft-ietf-trill-yang-oam-05, March 2017.

Acknowledgments

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

Carlos Pignataro, David Ball, Mahesh Jethanandani, Benoit Claise, Ladislav Lhotka, Jens Guballa, Yuji Tochio, Gregory Mirsky, Huub van Helvoort, Tom Taylor, Dapeng Liu, Mishael Wexler, and Adi Molkho contributed to and participated in the development of this document.

Contributors

Tissa Senevirathne
Consultant

Email: tsenevir@gmail.com

Norman Finn
CISCO Systems
510 McCarthy Blvd
Milpitas, CA 95035
United States of America

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
United States of America

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