Operations and Management Area Working Group
Internet-Draft
Intended status: Informational
Expires: 11 January 2024

L. M. Contreras
Telefonica
V. Lopez
Nokia

10 July 2023

A YANG Data Model for Network Diagnosis by scheduling sequences of OAM tests
draft-contreras-opsawg-scheduling-oam-tests-01

Abstract

This document defines a YANG data model for network diagnosis on-demand using Operations, Administration, and Maintenance (OAM) tests. This document defines both 'oam-unitary-test' and 'oam-test-sequence' data models to enable on-demand activation of network diagnosis procedures.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://vlopezalvarez.github.io/draft-contreras-opsawg-scheduling-oam-tests/draft-contreras-opsawg-scheduling-oam-tests.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-contreras-opsawg-scheduling-oam-tests/.

Discussion of this document takes place on the Operations and Management Area Working Group Working Group mailing list (mailto:opsawg@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/opsawg/. Subscribe at https://www.ietf.org/mailman/listinfo/opsawg/.

Source for this draft and an issue tracker can be found at https://github.com/vlopezalvarez/draft-contreras-opsawg-scheduling-oam-tests.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

This Internet-Draft will expire on 11 January 2024.

Copyright Notice

Table of Contents

1. Introduction

Operations, Administration, and Maintenance (OAM) tasks are fundamental functionalities of the network management. Given the emerging of data models and their utilization in Service Provider's management, the management of OAM tests represent also an area of interest for operators, which requires to be defined as a data model. OAM functionalities provide the means to identify and isolate faults, measure and report of performance. [RFC5860] defines the three main areas involved in OAM:

* Fault management, which allows network operators to quickly identify and isolate faults in the network. The OAM framework defines mechanisms for fault detection and isolation, such as continuity check, link trace, and loopback.

* Performance management enables monitoring network performance and diagnosing performance issues (i.e., degradation). Some of the measurements such as frame delay measurement, frame delay variation measurement, and frame loss measurement.

* Security management defines mechanisms to protect OAM communications from unauthorized access and tampering.

[RFC6428] defines several use cases for OAM tools, including:

* Continuity Check: This function verifies that a path exists between two points in a network and that the path is operational.

* Loopback: This function allows a device to loop back a received packet back to the sender for diagnostic purposes.

* Link Trace: This function allows a network operator to trace a path through a network from one device to another.

* Performance Monitoring: This function allows a network operator to monitor the performance of a network and to identify and diagnose performance issues.

* Security Management: This function allows a network operator to protect OAM communications from unauthorized access and tampering.

* Configuration Management: This function allows a network operator to manage the configuration of network devices.

[RFC8531], [RFC8532] and, [RFC8533] defined YANG models for OAM technologies. On the other hand, [RFC8531] defines a YANG data model for connection-oriented OAM protocols. The main aim of this document is to define a generic YANG data model that can be used to configure, control and monitor connection-oriented OAM protocols such as MPLS-TP OAM, PBB-TE OAM, and G.7713.1 OAM. [RFC8532] provides a generic YANG data model that can be used to configure, control and monitor connectionless OAM protocols such as BFD (Bidirectional Forwarding Detection), LBM (Loopback Messaging) and VCCV (Virtual Circuit Connectivity Verification). [RFC8533] provides a YANG data model that can be used to retrieve information related to OAM protocols such as Bidirectional Forwarding Detection (BFD), Loopback Messaging (LBM) and Virtual Circuit Connectivity Verification (VCCV).

[RFC8913] specifies a YANG data model for client and server implementations of the Two-Way Active Measurement Protocol (TWAMP).

Previous RFCs defined the parameters required for each of the different tests that are used in network elements today. This document covers how to use OAM for network-wide use cases. Following, some illustrative examples are presented.

The YANG data model resulting from this document will conform to the Network Management Datastore Architecture (NMDA) [RFC8342].

1.1. Terminology and Notations

This document assumes that the reader is familiar with the contents of [RFC7950], [RFC8345], [RFC8346] and [RFC8795].

Following terms are used for the representation of this data model.

* OAM unitary test: it is a set of parameters that define a type of OAM test to be invoked. As an example, it includes the type test, configuration parameters, and target results.

* OAM test sequence: it is a set of OAM unitary tests that are run based on a set of time constraints, number of repetitions, order, and reporting outputs.

1.2. Requirements Language

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

1.3. Prefix in Data Node Names

In this document, names of data nodes and other data model objects will be prefixed using the standard prefix associated with the corresponding YANG imported modules, as shown in the following table.

Prefix	Yang Module	Reference
oamut	ietf-oam-unitary-tests	RFCXXX
oamts	ietf-oam-test-sequence	RFCXXX
yang	ietf-yang-types	[RFC6991]

Table 1: Prefixes and corresponding YANG modules

RFC Editor Note: Please replace XXXX with the RFC number assigned to this document if the document becomes a RFC. Please remove this note in that case.

2. Network wide OAM use cases

2.1. Troubleshooting

After the detection of a problem in the network, OAM tests are performed to find the root cause for the detected issue. However, a problem detected can be caused by a variety of factors, such as a misconfiguration, hardware failure, or a software bug. OAM tests can help to find the root cause by testing specific components of the network and looking for anomalies or issues.

There are a variety of different OAM tests that can be executed depending on the nature of the scenario. For example, if the issue is related to a L2 capability, tests can be run to check the status of the path via Ethernet Linktrace and later run an Ethernet Loopback to a concrete network element. If these tests are correct, the operator may want to check the availability of the service or its performance.

Even though the troubleshooting process may be different depending on the problem detected, there are certain common procedures or logic that can be executed in order to narrow down the cause of the problem.

2.2. Birth certificate

The aim of a birth certificate process is to validate that all relevant parameters are correct for a specific network service. The birth certificate process is done once the configuration of the network elements is completed and they are ready for service.

If the birth certificate is successful, it means that the network service is functioning correctly and meets the requirements defined by the operator. The process requires running a set of OAM tests to verify that the service is performing as expected.

The set of OAM tests done as part of a birth certificate process depends on the network service that is tested. For example, if the service is a virtual private network (VPN) Two-Way Active Measurement Protocol (TWAMP) Light will be used, while if the service is an E-LINE, Ethernet CFM tests will be executed.

Once the birth certificate process has been completed and the OAM tests have been run, the test results are stored as part of the documentation process performed by the operator.

2.3. Proactive supervision

There are communication services that require to fulfill Service Level Agreements (SLAs). SLAs define performance parameters that the service must fulfill in order to meet the requirements of the customer or end user.

Proactive testing ensures the SLAs are met. Proactive supervision requires running tests on service components to identify and resolve issues before they impact the customer or end user, or to minimize the impact of the end user.

Proactive testing can be done via OAM tests. These tests can be run periodically at regular intervals depending on the specific SLA requirements and the network operator procedures. These procedures may require documenting the test results for future auditing processes with the customers.

2.4. Performance-based Path Routing

Path Computation Elements (PCEs) allow computing end-to-end paths in a network. PCEs are used to facilitate traffic engineering and can be used to optimize network performance, reduce congestion, and improve the overall user experience.

There are different algorithms to calculate a path in the network for some of them the PCE requires traffic engineering information. TE information includes data such as link metrics, bandwidth availability, and routing constraints. By using this information, the PCE can compute the optimal path for a particular service, taking into account its constraints and requirements. OAM techniques allow obtaining link metrics like delay and loss which can be used in the PCE algorithms.

3. Modelling the scheduling of OAM Tests

This document proposes two models: OAM unitary test and OAM test sequence models.

3.1. OAM unitary test

The OAM unitary test model encompasses parameters that define a specific type of OAM test to be performed. The YANG model includes a container named "oam-unitary-tests" that serves as a container for activating OAM unitary tests for network diagnosis procedures. Inside the container, there is a list called "oam-unitary-test" representing a list of specific OAM unitary tests. The list key is defined as "name", which provides a unique name for each test. Each OAM test in the list references a test type with its concrete parameters. The test types are out of the scope of this document. Moreover, each OAM unitary test has two temporal parameters: "period-of-time" and "recurrence". Both are imported from the "ietf-schedule" module from [I-D.draft-ma-opsawg-ucl-acl]. "period-of-time" identifies the period values that contain a precise period of time, while "recurrence" identifies the properties that contain a recurrence rule specification. "unitary-test-status" enumerates the state of the OAM unitary test.

Figure 1 contains the tree of the proposed model.

```
module: ietf-oam-unitary-test
  +--rw oam-unitary-tests
     +--rw oam-unitary-test* [name]
        +--rw name                    string
        +--rw (test-type)
        +--rw period-of-time
        |  +--rw (forms)?
        |     +--:(period-explicit)
        |     |  +--rw explicit-start?   yang:date-and-time
        |     |  +--rw explicit-end?     yang:date-and-time
        |     +--:(period-start)
        |        +--rw start?            yang:date-and-time
        |        +--rw duration?         duration
        +--rw recurrence
        |  +--rw freq           enumeration
        |  +--rw (recurrence-bound)?
        |  |  +--:(until)
        |  |  |  +--rw until?   union
        |  |  +--:(count)
        |  |     +--rw count?   uint32
        |  +--rw interval?      uint32
        |  +--rw bysecond*      uint32
        |  +--rw byminute*      uint32
        |  +--rw byhour*        uint32
        |  +--rw byday* [weekday]
        |  |  +--rw direction*   int32
        |  |  +--rw weekday      schedule:weekday
        |  +--rw bymonthday*    int32
        |  +--rw byyearday*     int32
        |  +--rw byyearweek*    int32
        |  +--rw byyearmonth*   uint32
        |  +--rw bysetpos*      int32
        |  +--rw wkst?          schedule:weekday
        +--rw unitary-test-status?   enumeration
```

Figure 1: OAM unitary test

The 'unitary-test-status' state machine is shown in Figure 2. The state machine includes the following states:

* "planned": The initial state where the test is planned.

* "configured": The state where the test is being configured.

* "ready": The state where the test is ready to be executed.

* "on-going": The state where the test is currently running.

* "stop": The state where the test is manually stopped.

* "error": The state where an error occurs during the test.

* "finished": The final state where the test is completed.

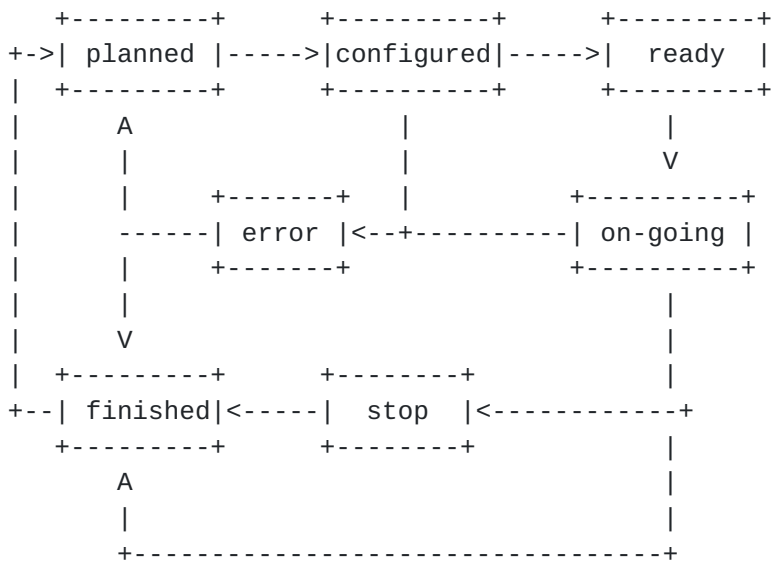

Figure 2: OAM unitary test state machine

3.2. OAM test sequence

The OAM test sequence model consists of a collection of OAM unitary tests that are executed based on specified time constraints, repetitions, ordering, and reporting outputs. These sequences provide a structured approach to running multiple OAM tests in a coordinated manner. Each OAM test sequence references a OAM unitary test type with its concrete parameters. Each OAM test sequence has two temporal parameters: "period-of-time" and "recurrence". Both are imported from the "ietf-schedule" module from [I-D.draft-ma-opsawg-ucl-acl]. "period-of-time" identifies the period values that contain a precise period of time, while "recurrence" identifies the properties that contain a recurrence rule specification. "unitary-test-status" enumerates the state of the OAM test. Finally, "test-sequence-status" shows the state of the OAM test sequence.

An example of the proposed structure Figure 3.

```
module: ietf-oam-test-sequence
  +--rw oam-test-sequence
     +--rw test-sequence* [name]
        +--rw name                    string
        +--rw test-ref* [name]
        |  +--rw name              string
        |  +--rw (test-type)
        |  +--rw numexecutions?   uint32
        +--rw period-of-time
        |  +--rw (forms)?
        |     +--:(period-explicit)
        |     |  +--rw explicit-start?   yang:date-and-time
        |     |  +--rw explicit-end?     yang:date-and-time
        |     +--:(period-start)
        |        +--rw start?            yang:date-and-time
        |        +--rw duration?         duration
        +--rw recurrence
        |  +--rw freq           enumeration
        |  +--rw (recurrence-bound)?
        |  |  +--:(until)
        |  |  |  +--rw until?   union
        |  |  +--:(count)
        |  |     +--rw count?   uint32
        |  +--rw interval?      uint32
        |  +--rw bysecond*      uint32
        |  +--rw byminute*      uint32
        |  +--rw byhour*        uint32
        |  +--rw byday* [weekday]
        |  |  +--rw direction*   int32
        |  |  +--rw weekday      schedule:weekday
        |  +--rw bymonthday*    int32
        |  +--rw byyearday*     int32
        |  +--rw byyearweek*    int32
        |  +--rw byyearmonth*   uint32
        |  +--rw bysetpos*      int32
        |  +--rw wkst?          schedule:weekday
        +--ro sequence?   enumeration
```

Figure 3: OAM test sequence

The 'test-sequence-status' state machine is shown in Figure 4. The state machine includes the following states:

* "planned": The initial state where the test is planned.

* "configured": The state where the test is being configured.

* "ready": The state where the test is ready to be executed.

* "on-going": The state where the test is currently running.

* "stop": The state where the test is manually stopped.

* "success": The success state when all unitary tests were successful.

* "failure": The success state when One or more tests in the sequence got an error.

* "error": The state where an error occurs during the test.

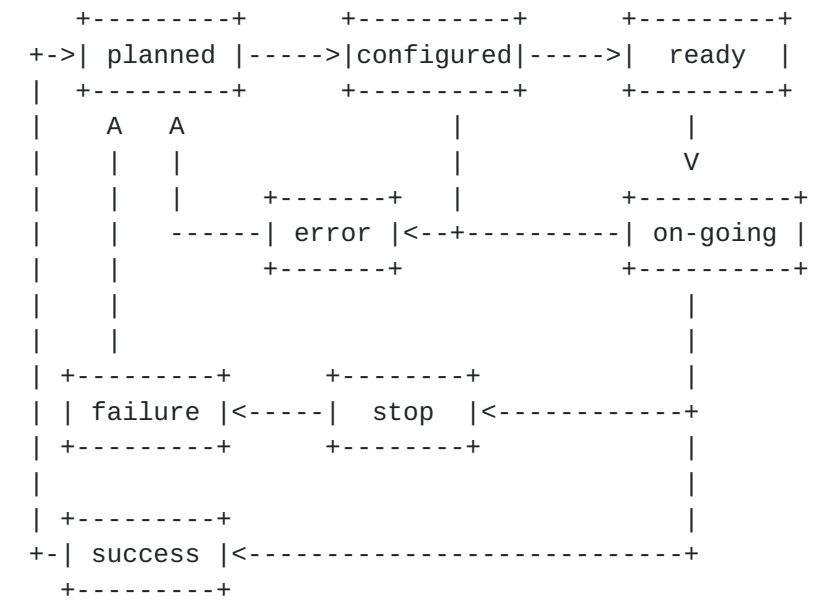

Figure 4: OAM unitary test state machine

4. YANG Data Model for Scheduling OAM Tests

4.1. YANG Model Overview

This document proposes two models: OAM unitary test and OAM test sequence. OAM unitary test is a set of parameters that define a type of OAM test to be invoked. As an example, it includes the type test, configuration parameters, and target results. The OAM test sequences are a set of OAM unitary tests that are run based on a set of time constraints, number of repetitions, order, and reporting outputs.

4.2. YANG Model for Scheduling OAM Unitary Test

```
   <CODE BEGINS>
   module ietf-oam-unitary-test {
     yang-version 1.1;
     namespace "urn:ietf:params:xml:ns:yang:ietf-oam-unitary-test";
     prefix "oamut";

     // Import OAM models from RFCs RFC8531, RFC8532 and RFC8533
     import ietf-schedule { prefix "schedule"; } // reference draft-ma-opsawg-
ucl-acl

     organization
       "IETF OPSAWG (Operations and Management Area Working Group)";

     contact
       "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
        WG List:  <mailto:opsawg@ietf.org>
        Author: Luis Miguel Contreras Murillo
             <luismiguel.contrerasmurillo@telefonica.com>
        Author: Victor Lopez
             <victor.lopez@nokia.com>";
     description
       "This module defines the 'ietf-oam-unitary-test' YANG model for
activation of network diagnosis procedures.

       Copyright (c) 2023 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject
       to the license terms contained in, the Revised BSD License
       set forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
         (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.";

     // RFC Ed.: update the date below with the date of RFC
     // publication and remove this note.
     // RFC Ed.: replace XXXX with actual RFC number and remove
     // this note.

     revision "2023-07-10" {
       description
         "Initial version";
       reference
         "RFCXXXX: A YANG Data Model for Network Diagnosis by scheduling
sequences of OAM tests"; // Update with the correct RFC number when assigned
```

```
}
```

```
     grouping oam-unitary-test {
       description
           "This grouping is defined for OAM unitary test for network diagnosis
procedures.";

       leaf name {
         type string;
         description
           "Name for the test.";
       }

       choice test-type {
         mandatory true;
         description
           "Choose the type of test.";
           // Import OAM models from RFCs RFC8531, RFC8532 and RFC8533
       }
     }

     container oam-unitary-tests {
       description
         "Container for OAM unitary tests activation for network diagnosis
procedures.";

       list oam-unitary-test {
         key name;
         description
           "List of OAM unitary tests activation for network diagnosis
procedures.";

         uses oam-unitary-test;

         uses schedule:period;

         uses schedule:recurrence;

         leaf unitary-test-status {
           type enumeration {
             enum "planned" {
               description "The test is planned.";
             }
             enum "configure" {
               description "The test is configured.";
             }
             enum "ready" {
               description "The test status is ready.";
             }
             enum "ongoing" {
               description "The test is ongoing.";
```

```
            }
            enum "stop" {
```

```
              description "The test is stopped.";
            }
            enum "finish" {
              description "The test is finished.";
            }
            enum "error" {
              description "The test has an error.";
            }
          }
          description "Status of the test.";
        }
      }
    }
  }
  <CODE ENDS>
```

4.3. YANG Model for OAM Test Sequence

```
  <CODE BEGINS>
  module ietf-oam-test-sequence {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-oam-test-sequence";
    prefix "oamts";

    import ietf-oam-unitary-test {
      prefix "oamut";
      // Update the reference with the correct RFC number or other reference
when assigned
      //reference "RFCXXXX";
    }

    import ietf-schedule { prefix "schedule"; } // reference draft-ma-opsawg-
ucl-acl

    organization
      "IETF OPSAWG (Operations and Management Area Working Group)";

    contact
      "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
       WG List:  <mailto:opsawg@ietf.org>
       Author: Luis Miguel Contreras Murillo
            <luismiguel.contrerasmurillo@telefonica.com>
       Author: Victor Lopez
            <victor.lopez@nokia.com>";
    description
      "This module defines the 'oam-unitary-test' YANG model for activation of
network diagnosis procedures.
```

```
       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject
       to the license terms contained in, the Revised BSD License
       set forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
         (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.";

     // RFC Ed.: update the date below with the date of RFC
     // publication and remove this note.
     // RFC Ed.: replace XXXX with actual RFC number and remove
     // this note.

     revision "2023-07-10" {
       description "Initial version";
       reference "RFCXXXX"; // Update with the correct RFC number when assigned
     }

     // Data model definition

     container oam-test-sequence {
       description "Container for executing a sequence of ietf-oam-unitary-
tests N times.";

       list test-sequence {
         key "name";
         description "List of test sequences.";

         leaf name {
           type string;
           description "Unique name for the test sequence.";
         }

         list test-ref {
           key "name";
           description "References to the ietf-oam-unitary-tests.";

           uses "oamut:oam-unitary-test";

           leaf numexecutions {
             type uint32;
             default 1;
             description "Number of times the test sequence should be
executed.";
           }
         }
```

```
        uses schedule:period;

        uses schedule:recurrence;

        leaf test-squence-status {
          type enumeration {
            enum "planned" {
              description "The test sequence is planned.";
            }
            enum "success" {
              description "All tests in the sequence were successful.";
            }
            enum "failure" {
              description "One or more tests in the sequence failed.";
            }
            enum "ongoing" {
              description "The test sequence status is ongoing.";
            }
            enum "unknown" {
              description "The test sequence status is unknown.";
            }
            enum "stop" {
              description "The test sequenceis stopped.";
            }
            enum "finish" {
              description "The test sequence is finished.";
            }
          }
          config false;
          description "Status of the test sequence execution.";
        }
      }
    }
  }
  <CODE ENDS>
```

5. Security Considerations

The YANG module targeted 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 [RFC5246].

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

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

6. IANA Considerations

TBC

7. Implementation Status

This section will be used to track the status of the implementations of the model. It is aimed at being removed if the document becomes RFC.

8. Normative References

[I-D.draft-ma-opsawg-ucl-acl]
Ma, Q., Wu, Q., Boucadair, M., and D. King, "A Policy-based Network Access Control", Work in Progress, Internet-Draft, draft-ma-opsawg-ucl-acl-03, 7 June 2023, <https://datatracker.ietf.org/doc/html/draft-ma-opsawg-ucl-acl-03>.

[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/rfc/rfc2119>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/rfc/rfc5246>.

[RFC5860] Vigoureux, M., Ed., Ward, D., Ed., and M. Betts, Ed., "Requirements for Operations, Administration, and Maintenance (OAM) in MPLS Transport Networks", RFC 5860, DOI 10.17487/RFC5860, May 2010, <https://www.rfc-editor.org/rfc/rfc5860>.

[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/rfc/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/rfc/rfc6242>.

[RFC6428] Allan, D., Ed., Swallow, G., Ed., and J. Drake, Ed., "Proactive Connectivity Verification, Continuity Check, and Remote Defect Indication for the MPLS Transport Profile", RFC 6428, DOI 10.17487/RFC6428, November 2011, <https://www.rfc-editor.org/rfc/rfc6428>.

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

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

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/rfc/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/rfc/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/rfc/rfc8174>.

[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/rfc/rfc8342>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <https://www.rfc-editor.org/rfc/rfc8345>.

[RFC8346] Clemm, A., Medved, J., Varga, R., Liu, X., Ananthakrishnan, H., and N. Bahadur, "A YANG Data Model for Layer 3 Topologies", RFC 8346, DOI 10.17487/RFC8346, March 2018, <https://www.rfc-editor.org/rfc/rfc8346>.

[RFC8531] Kumar, D., Wu, Q., and Z. Wang, "Generic YANG Data Model for Connection-Oriented Operations, Administration, and Maintenance (OAM) Protocols", RFC 8531, DOI 10.17487/RFC8531, April 2019, <https://www.rfc-editor.org/rfc/rfc8531>.

[RFC8532] Kumar, D., Wang, Z., 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/rfc/rfc8532>.

[RFC8533] Kumar, D., Wang, M., Wu, Q., Ed., Rahman, R., and S. Raghavan, "A YANG Data Model for Retrieval Methods for the Management of Operations, Administration, and Maintenance (OAM) Protocols That Use Connectionless Communications", RFC 8533, DOI 10.17487/RFC8533, April 2019, <https://www.rfc-editor.org/rfc/rfc8533>.

[RFC8795] Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Gonzalez de Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", RFC 8795, DOI 10.17487/RFC8795, August 2020, <https://www.rfc-editor.org/rfc/rfc8795>.

[RFC8913] Civil, R., Morton, A., Rahman, R., Jethanandani, M., and K. Pentikousis, Ed., "Two-Way Active Measurement Protocol (TWAMP) YANG Data Model", RFC 8913, DOI 10.17487/RFC8913, November 2021, <https://www.rfc-editor.org/rfc/rfc8913>.

Acknowledgments

TODO acknowledge.

Authors' Addresses

Luis M. Contreras
Telefonica
Email: luismiguel.contrerasmurillo@telefonica.com

Victor Lopez
Nokia
Email: victor.lopez@nokia.com