Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 31, 2017

G. Mirsky
ZTE Corp.
A. Pan
Ericsson

February 27, 2017

Two-Way Active Measurement Protocol (TWAMP) Light Data Model
draft-mirsky-ippm-twamp-light-yang-06

Abstract

This document specifies the data model for implementations of Session-Sender and Session-Reflector for Two-Way Active Measurement Protocol (TWAMP) Light mode using YANG.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 31, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Conventions used in this document 2
1.1.1. Requirements Language 2
2. Scope, Model, and Applicability 3
2.1. Data Model Parameters 3
2.1.1. Session-Sender 3
2.1.2. Session-Reflector 4
3. Data Model 4
3.1. Tree Diagram 4
3.2. YANG Module 8
4. IANA Considerations 19
5. Security Considerations 19
6. Acknowledgements 19
7. References 19
7.1. Normative References 19
7.2. Informative References 20
Authors' Addresses 21

1. Introduction

The Two-Way Active Measurement Protocol (TWAMP) [RFC5357] can be used to measure performance parameters of IP networks such as latency, jitter, and packet loss by sending test packets and monitoring their experience in the network. The [RFC5357] defines two protocols, TWAMP Control and TWAMP Test, and a profile of TWAMP Test, TWAMP Light. The TWAMP Light is known to have many implementations though no common management framework being defined, thus leaving some aspects of test packet processing to interpretation. The goal of this document is to collect analyze these variations; describe common model while allowing for extensions in the future. This document defines such a TWAMP data model and specifies it formally using the YANG data modeling language [RFC6020].

1.1. Conventions used in this document

1.1.1. 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].

2. Scope, Model, and Applicability

The scope of this document includes model of the TWAMP Light as defined in Appendix I of [RFC5357]. This mode of TWAMP Light will be referred in this document as Stateless. Another mode, where the Session-Reflector is aware of the state of the TWAMP test session and thus can independently count reflected test packets, referred as Stateful. This document benefits from earlier attempt to define TWAMP MIB in [I-D.elteto-ippm-twamp-mib] and from TWAMP YANG model defined in [I-D.ietf-ippm-twamp-yang].

Figure 1 updates TWAMP-Light reference model presented in Appendix I [RFC5357] for the scenario when instantiation of a TWAMP-Test session between Session-Sender and Session-Reflector controlled by communication between a Configuration Client as a manager and Configuration Servers as agents of the configuration session.

```
      o-------------------------------------------------------o
      |                      Config client                    |
      o-------------------------------------------------------o
            ||                                          ||
            ||            NETCONF/RESTCONF              ||
            ||                                          ||
   o-------------------o                       o-------------------o
   |   Config server   |                       |   Config server   |
   |                   |                       |                   |
   +-------------------+                       +-------------------+
   |  Session-Sender   |  <--- TWAMP-Test ---> | Session-Reflector |
   +-------------------+                       +-------------------+
```

Figure 1: TWAMP Light Reference Model

2.1. Data Model Parameters

This section describes all the parameters of the the TWAMP-Light data model.

2.1.1. Session-Sender

The twamp-light-session-sender container holds items that are related to the configuration of the TWAMP-Light Session-Sender logical entity.

The twamp-light-session-sender-state container holds information about the state of the particular TWAMP-Light test session.

RPCs twamp-sender-start and twamp-sender-stop respectively start and stop the referenced by session-id TWAMP-Light test session.

2.1.2. Session-Reflector

The twamp-light-session-reflector container holds items that are related to the configuration of the TWAMP-Light Session-Reflector logical entity.

The twamp-light-session-reflector-state container holds Session-Reflector state data for the particular TWAMP-Light test session.

3. Data Model

Creating TWAMP-Light data model presents number of challenges and among them is identification of a test-session at Session-Reflector. A Session-Reflector MAY require only as little as its IP and UDP port number in received TWAMP-Test packet to spawn new test session. More so, to test processing of Class-of-Service along the same route in Equal Cost Multi-Path environment Session-Sender may run TWAMP test sessions concurrently using the same source IP address, source UDP port number, destination IP address, and destination UDP port number. Thus the only parameter that can be used to differentiate these test sessions would be DSCP value. The DSCP field may get re-marked along the path and without use of [RFC7750] that will go undetected, but by using five-tuple instead of four-tuple as a key we can ensure that TWAMP test packets that are considered as different test sessions follow the same path even in ECMP environments.

3.1. Tree Diagram

```
module: ietf-twamp-light
    +--rw twamp-light
    |  +--rw twamp-light-session-sender {session-sender-light}?
    |  |  +--rw sender-light-enable?   enable
    |  |  +--rw test-session* [session-id]
    |  |     +--rw session-id               uint32
    |  |     +--rw test-session-enable?     enable
    |  |     +--rw number-of-packets?       uint32
    |  |     +--rw packet-padding-size?     uint32
    |  |     +--rw interval?                uint32
    |  |     +--rw sender-ip                inet:ip-address
    |  |     +--rw sender-udp-port          inet:port-number
    |  |     +--rw reflector-ip             inet:ip-address
    |  |     +--rw reflector-udp-port       inet:port-number
    |  |     +--rw dscp?                    inet:dscp
```

```
    |  |     +--rw authentication-params! {twamp-light-authentication}?
    |  |        +--rw key-chain?   kc:key-chain-ref
    |  +--rw twamp-light-session-reflector {session-reflector-light}?
    |     +--rw reflector-light-enable?       enable
    |     +--rw ref-wait?                     uint32
    |     +--rw reflector-light-mode-state?   enumeration
    |     +--rw dscp-handling-mode?           enumeration
    |     +--rw test-session* [session-id]
    |        +--rw session-id               uint32
    |        +--rw sender-ip                inet:ip-address
    |        +--rw sender-udp-port          inet:port-number
    |        +--rw reflector-ip             inet:ip-address
    |        +--rw reflector-udp-port       inet:port-number
    |        +--rw dscp?                    inet:dscp
    |        +--rw authentication-params! {twamp-light-authentication}?
    |           +--rw key-chain?   kc:key-chain-ref
    +--ro twamp-light-state
       +--ro twamp-light-session-sender-state {session-sender-light}?
       |  +--ro test-session-state* [session-id]
       |     +--ro session-id              uint32
       |     +--ro sender-session-state?   enumeration
       |     +--ro current-stats
       |     |  +--ro start-time                yang:date-and-time
       |     |  +--ro number-of-packets?        uint32
       |     |  +--ro packet-padding-size?      uint32
       |     |  +--ro interval?                 uint32
       |     |  +--ro duplicate-packets?        uint32
       |     |  +--ro reordered-packets?        uint32
       |     |  +--ro loss-packets?             uint32
       |     |  +--ro sender-ip                 inet:ip-address
       |     |  +--ro sender-udp-port           inet:port-number
       |     |  +--ro reflector-ip              inet:ip-address
       |     |  +--ro reflector-udp-port        inet:port-number
       |     |  +--ro dscp?                     inet:dscp
       |     |  +--ro sent-packets?             uint32
       |     |  +--ro rcv-packets?              uint32
       |     |  +--ro sent-packets-error?       uint32
       |     |  +--ro rcv-packets-error?        uint32
       |     |  +--ro last-sent-seq?            uint32
       |     |  +--ro last-rcv-seq?             uint32
       |     |  +--ro two-way-delay
       |     |  |  +--ro delay
       |     |  |  |  +--ro min?   yang:gauge32
       |     |  |  |  +--ro max?   yang:gauge32
       |     |  |  |  +--ro avg?   yang:gauge32
       |     |  |  +--ro delay-variation
       |     |  |     +--ro min?   uint32
       |     |  |     +--ro max?   uint32
```

```
        |     |  |     +--ro avg?   uint32
        |     |  +--ro one-way-delay-far-end
        |     |  |  +--ro delay
        |     |  |  |  +--ro min?   yang:gauge32
        |     |  |  |  +--ro max?   yang:gauge32
        |     |  |  |  +--ro avg?   yang:gauge32
        |     |  |  +--ro delay-variation
        |     |  |     +--ro min?   uint32
        |     |  |     +--ro max?   uint32
        |     |  |     +--ro avg?   uint32
        |     |  +--ro one-way-delay-near-end
        |     |     +--ro delay
        |     |     |  +--ro min?   yang:gauge32
        |     |     |  +--ro max?   yang:gauge32
        |     |     |  +--ro avg?   yang:gauge32
        |     |     +--ro delay-variation
        |     |        +--ro min?   uint32
        |     |        +--ro max?   uint32
        |     |        +--ro avg?   uint32
        |     +--ro history-stats* [id]
        |        +--ro id                        uint32
        |        +--ro end-time                  yang:date-and-time
        |        +--ro number-of-packets?        uint32
        |        +--ro packet-padding-size?      uint32
        |        +--ro interval?                 uint32
        |        +--ro duplicate-packets?        uint32
        |        +--ro reordered-packets?        uint32
        |        +--ro loss-packets?             uint32
        |        +--ro sender-ip                 inet:ip-address
        |        +--ro sender-udp-port           inet:port-number
        |        +--ro reflector-ip              inet:ip-address
        |        +--ro reflector-udp-port        inet:port-number
        |        +--ro dscp?                     inet:dscp
        |        +--ro sent-packets?             uint32
        |        +--ro rcv-packets?              uint32
        |        +--ro sent-packets-error?       uint32
        |        +--ro rcv-packets-error?        uint32
        |        +--ro last-sent-seq?            uint32
        |        +--ro last-rcv-seq?             uint32
        |        +--ro two-way-delay
        |        |  +--ro delay
        |        |  |  +--ro min?   yang:gauge32
        |        |  |  +--ro max?   yang:gauge32
        |        |  |  +--ro avg?   yang:gauge32
        |        |  +--ro delay-variation
        |        |     +--ro min?   uint32
        |        |     +--ro max?   uint32
        |        |     +--ro avg?   uint32
```

```
      |        +--ro one-way-delay-far-end
      |        |  +--ro delay
      |        |  |  +--ro min?   yang:gauge32
      |        |  |  +--ro max?   yang:gauge32
      |        |  |  +--ro avg?   yang:gauge32
      |        |  +--ro delay-variation
      |        |     +--ro min?   uint32
      |        |     +--ro max?   uint32
      |        |     +--ro avg?   uint32
      |        +--ro one-way-delay-near-end
      |           +--ro delay
      |           |  +--ro min?   yang:gauge32
      |           |  +--ro max?   yang:gauge32
      |           |  +--ro avg?   yang:gauge32
      |           +--ro delay-variation
      |              +--ro min?   uint32
      |              +--ro max?   uint32
      |              +--ro avg?   uint32
      +--ro twamp-light-session-reflector-state {session-reflector-light}?
         +--ro reflector-light-admin-status    boolean
         +--ro test-session-state* [session-id]
            +--ro session-id            uint32
            +--ro sent-packets?         uint32
            +--ro rcv-packets?          uint32
            +--ro sent-packets-error?   uint32
            +--ro rcv-packets-error?    uint32
            +--ro last-sent-seq?        uint32
            +--ro last-rcv-seq?         uint32
            +--ro sender-ip             inet:ip-address
            +--ro sender-udp-port       inet:port-number
            +--ro reflector-ip          inet:ip-address
            +--ro reflector-udp-port    inet:port-number
            +--ro dscp?                 inet:dscp
            +--ro one-way-delay
               +--ro delay
               |  +--ro min?   yang:gauge32
               |  +--ro max?   yang:gauge32
               |  +--ro avg?   yang:gauge32
               +--ro delay-variation
                  +--ro min?   uint32
                  +--ro max?   uint32
                  +--ro avg?   uint32

  rpcs:
    +---x twamp-sender-start
    |  +---w input
    |     +---w session-id    uint32
    +---x twamp-sender-stop
```

```
       +---w input
          +---w session-id    uint32
```

3.2. YANG Module

```
 <CODE BEGINS> file "ietf-twamp-light@2017-02-27.yang"

 module ietf-twamp-light {
    namespace "urn:ietf:params:xml:ns:yang:ietf-twamp-light";
     //namespace need to be assigned by IANA
    prefix "ietf-twamp-light";

    import ietf-inet-types {
      prefix inet;
    }
    import ietf-yang-types {
      prefix yang;
    }
    import ietf-key-chain {
      prefix kc;
    }

    organization
      "IETF IPPM (IP Performance Metrics) Working Group";

    contact
      "draft-mirsky-ippm-twamp-light-yang@tools.ietf.org";

    description "TWAMP Light Data Model";

    revision "2017-02-27" {
      description "06 version. Appendix I RFC 5357 is covered.";
      reference "RFC 5357";
    }

    feature session-sender-light {
      description "This feature relates to the device functions as the
      TWAMP Light Session-Sender";
    }

    feature session-reflector-light {
      description "This feature relates to the device functions as the
      TWAMP Light Session-Reflector";
    }

    feature twamp-light-authentication {
```

```
      description "TWAMP Light authentication supported";
    }

    typedef enable {
       type boolean;
       description "enable";
    }

    grouping maintenance-statistics {
      description "Maintenance statistics grouping";
      leaf sent-packets {
        type uint32;
        description "Packets sent";
      }
      leaf rcv-packets {
        type uint32;
        description "Packets received";
      }
      leaf sent-packets-error {
        type uint32;
        description "Packets sent error";
      }
      leaf rcv-packets-error {
        type uint32;
        description "Packets received error";
      }
      leaf last-sent-seq {
        type uint32;
        description "Last sent sequence number";
      }
      leaf last-rcv-seq {
        type uint32;
        description "Last received sequence number";
      }
    }

    grouping delay-statistics {
      description "delay statistics grouping";
      container delay {
        description "Packets transmitted delay";
                  leaf min {
                    type yang:gauge32;
                    units microseconds;
                    description "Min of Packets transmitted delay";
                  }
                  leaf max {
                     type yang:gauge32;
                     units microseconds;
```

```
                    description "Max of Packets transmitted delay";
                 }
                 leaf avg {
                    type yang:gauge32;
                    units microseconds;
                    description "Avg of Packets transmitted delay";
                 }
      }

          container delay-variation {
                 description "Packets transmitted delay variation";
                 leaf min {
                   type uint32;
                   units microseconds;
                   description "Min of Packets
                   transmitted delay variation";
                 }
                 leaf max {
                   type uint32;
                   units microseconds;
                   description "Max of Packets
                   transmitted delay variation";
                 }
                 leaf avg {
                   type uint32;
                   units microseconds;
                   description "Avg of Packets
                   transmitted delay variation";
                 }
           }
    }

    grouping session-light-parameters {
      description "Parameters common among
      Session-Sender and Session-Reflector";
      leaf sender-ip {
        type inet:ip-address;
        mandatory true;
        description "Sender IP address";
      }
      leaf sender-udp-port {
        type inet:port-number {
          range "49152..65535";
        }
        mandatory true;
        description "Sender UDP port number";
      }
      leaf reflector-ip {
```

```
        type inet:ip-address;
        mandatory true;
        description "Reflector IP address";
      }
      leaf reflector-udp-port {
        type inet:port-number{
          range "49152..65535";
        }
        mandatory true;
        description "Reflector UDP port number";
      }
      leaf dscp {
        type inet:dscp;
        default "0";
        description "The DSCP value to be placed in the header of
        TWAMP UDP test packets generated by the Session-Sender.
        Whether Session-Reflector uses this value depends upon its
        local configuration";
      }
    }

    grouping session-light-auth-params {
       description "Grouping for TWAMP Light authentication parameters";
       container authentication-params {
          if-feature twamp-light-authentication;
          presence "Enables TWAMP Light authentication";
          description "Parameters for TWAMP Light authentication";
          leaf key-chain {
             type kc:key-chain-ref;
             description "Name of key-chain";
          }
       }
    }

    /*Configuration Data*/
    container twamp-light {
      description "Top level container for TWAMP-Light configuration";

     container twamp-light-session-sender {
        if-feature session-sender-light;
        description "TWAMP-Light Session-Sender container";

        leaf sender-light-enable {
          type enable;
          default "true";
          description "Whether this network element is enabled to
          act as TWAMP-Light Sender";
        }
```

```
        list test-session {
          key "session-id";
                unique "sender-ip sender-udp-port reflector-ip"
          +" reflector-udp-port dscp";
          description "This structure is a container of test session
          managed objects";

          leaf session-id {
            type uint32;
            description "Session ID";
          }

          leaf test-session-enable {
            type enable;
            default "true";
            description "Whether this TWAMP Test session is enabled";
          }

          leaf number-of-packets {
            type uint32;
            description "The overall number of UDP test packets to be
              transmitted by the sender for this test session";
          }

          leaf packet-padding-size {
            type uint32;
            default 27;
            description "Size of the Packet Padding. Suggested to run
            Path MTU Discovery to avoid packet fragmentation in IPv4
            and packet blackholing in IPv6";
          }

          leaf interval  {
            type uint32;
            description "Time interval between transmission of two

            consecutive packets in the test session";
          }

          uses session-light-parameters;
          uses session-light-auth-params;
        }
      }

      container twamp-light-session-reflector {
        if-feature session-reflector-light;
        description "TWAMP-Light Session-Reflector container";
          leaf reflector-light-enable {
```

```
            type enable;
            default "true";
            description "Whether this network element is enabled to
            act as TWAMP-Light Reflector";
          }

          leaf ref-wait {
          type uint32 {
            range 1..604800;
          }
          units seconds;
          default 900;
          description "REFWAIT(TWAMP test session timeout in seconds),
          the default value is 900";
        }

        leaf reflector-light-mode-state {
          type enumeration {
            enum stateful {
              description "When the Session-Reflector Light is
              stateful, i.e. is aware of test session state";
            }
            enum stateless {
              description "When the Session-Reflector is stateless
              and uses sequence number of the received TWAMP-Test
              packet as its own";
            }
          }
          default stateless;
          description "The state of the mode of the TWAMP-Light
          Session-Reflector";
         }

         leaf dscp-handling-mode {
           type enumeration {
             enum copy-received-value {
               description "Use DSCP value copied from received TWAMP
               test packet of the test session";
               }
               enum use-configured-value {
                 description "Use DSCP value configured for this test
                 session on the Session-Reflector";
                 }
             }
             default copy-received-value;
             description "Session-Reflector handling of DSCP:
                 - use value copied from received TWAMP-Test packet;
                 - use value explicitly configured";
```

```
        }

        list test-session {
          key "session-id";
                unique "sender-ip sender-udp-port reflector-ip"
          +" reflector-udp-port dscp";
          description "This structure is a container of test session
          managed objects";

        leaf session-id {
          type uint32;
          description "Session ID";
        }

        uses session-light-parameters;
        uses session-light-auth-params;
       }
     }
   }

  /*Operational state data nodes*/
  container twamp-light-state{
    config "false";
    description "Top level container for TWAMP-Light state data";

    container twamp-light-session-sender-state {
      if-feature session-sender-light;
      description "Session-Sender container for state data";
      list test-session-state{
        key "session-id";
        description "This structure is a container of test session
        managed objects";

        leaf session-id {
          type uint32;
          description "Session ID";
        }

        leaf sender-session-state {
          type enumeration {
            enum active {
              description "Test session is active";
            }
            enum ready {
              description "Test session is idle";
            }
          }
          description "State of the particular TWAMP-Light test
```

```
          session at the sender";
        }

        container current-stats {
          description
          "This container contains the results for the current
           Measurement Interval in a Measurement session ";
           leaf start-time {
             type yang:date-and-time;
             mandatory true;
             description
             "The time that the current Measurement Interval started";
           }
           leaf number-of-packets {
             type uint32;
             description "The overall number of UDP test packets to be
             transmitted by the sender for this test session";
           }

           leaf packet-padding-size {
             type uint32;
             default 27;
             description "Size of the Packet Padding. Suggested to run
             Path MTU Discovery  to avoid packet fragmentation in IPv4
             and packet backholing in IPv6";
           }

           leaf interval  {
             type uint32;
            description "Time interval between transmission of two
            consecutive packets in the test session";
           }

           leaf duplicate-packets  {
             type uint32;
            description "Duplicate packets";
           }
           leaf reordered-packets  {
             type uint32;
            description "Reordered packets";
           }
           leaf loss-packets  {
             type uint32;
            description "Loss packets";
           }
           uses session-light-parameters;
           uses maintenance-statistics;
```

```
           container two-way-delay{
             description "two way delay result of the test session";
             uses delay-statistics;
           }
           container one-way-delay-far-end{
             description "one way delay far end of the test session";
             uses delay-statistics;
           }
           container one-way-delay-near-end{
             description "one way delay near end of the test session";
             uses delay-statistics;
           }
        }

        list history-stats {
          key id;
          description
          "This container contains the results for the history
           Measurement Interval in a Measurement session ";
           leaf id {
             type uint32;
             description
             "The identifier for the Measurement Interval within this
              session";
           }
           leaf end-time {
             type yang:date-and-time;
             mandatory true;
             description
             "The time that the Measurement Interval ended";
           }
           leaf number-of-packets {
             type uint32;
             description "The overall number of UDP test packets to be
             transmitted by the sender for this test session";
           }

           leaf packet-padding-size {
             type uint32;
             default 27;
             description "Size of the Packet Padding. Suggested to run
             Path MTU Discovery  to avoid packet fragmentation in IPv4
             and packet blackholing in IPv6";
           }

           leaf interval  {
             type uint32;
            description "Time interval between transmission of two
```

```
           consecutive packets in the test session";
          }
          leaf duplicate-packets  {
            type uint32;
           description "Duplicate packets";
          }
          leaf reordered-packets  {
            type uint32;
           description "Reordered packets";
          }
          leaf loss-packets  {
            type uint32;
           description "Loss packets";
          }

          uses session-light-parameters;
          uses maintenance-statistics;

          container two-way-delay{
            description "two way delay result of the test session";
            uses delay-statistics;
          }
          container one-way-delay-far-end{
            description "one way delay far end of the test session";
            uses delay-statistics;
          }
          container one-way-delay-near-end{
            description "one way delay near end of the test session";
            uses delay-statistics;
          }
        }
      }
    }

    container twamp-light-session-reflector-state {
      if-feature session-reflector-light;
      description "TWAMP-Light Session-Reflector container for
      state data";
      leaf reflector-light-admin-status {
        type boolean;
        mandatory "true";
        description "Whether this network element is enabled to
        act as TWAMP-Light Reflector";
      }

      list test-session-state {
        key "session-id";
        description "This structure is a container of test session
```

```
        managed objects";

        leaf session-id {
          type uint32;
          description "Session ID";
        }

        uses maintenance-statistics;
        uses session-light-parameters;

        container one-way-delay {
          description "one way delay of the test session";
          uses delay-statistics;
        }
      }
    }
  }

  rpc twamp-sender-start {
    description
          "start the configured sender session";
    input {
          leaf session-id {
            type uint32;
            mandatory true;
            description
                  "The session to be started";
          }
    }
  }

  rpc twamp-sender-stop {
    description
          "stop the configured sender session";
    input {
          leaf session-id {
            type uint32;
            mandatory true;
            description
                  "The session to be stopped";
          }
    }
  }
 }

  <CODE ENDS>
```

4. IANA Considerations

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

URI: urn:ietf:params:xml:ns:yang:ietf-twamp-light

Registrant Contact: The IPPM WG of the IETF.

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-twamp-light

namespace: urn:ietf:params:xml:ns:yang:ietf-twamp-light

prefix: twamp

reference: RFC XXXX

5. Security Considerations

The configuration, state, action data defined in this document may be accessed via the NETCONF protocol [RFC6241]. SSH [RFC6242] is mandatory secure transport that is the lowest NETCONF layer. The NETCONF access control model [RFC6536] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

But, in general, this TWAMP Light YANG module does not change any underlying security issues that already may exist in [I-D.elteto-ippm-twamp-mib].

6. Acknowledgements

7. References

7.1. Normative References

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

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

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <http://www.rfc-editor.org/info/rfc5357>.

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

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

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

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

[RFC7750] Hedin, J., Mirsky, G., and S. Baillargeon, "Differentiated Service Code Point and Explicit Congestion Notification Monitoring in the Two-Way Active Measurement Protocol (TWAMP)", RFC 7750, DOI 10.17487/RFC7750, February 2016, <http://www.rfc-editor.org/info/rfc7750>.

7.2. Informative References

[I-D.elteto-ippm-twamp-mib]
Elteto, T. and G. Mirsky, "Two-Way Active Measurement Protocol (TWAMP) Management Information Base (MIB)", draft-elteto-ippm-twamp-mib-01 (work in progress), January 2014.

[I-D.ietf-ippm-twamp-yang]
Civil, R., Morton, A., Rahman, R., Jethanandani, M., and K. Pentikousis, "Two-Way Active Measurement Protocol (TWAMP) Data Model", draft-ietf-ippm-twamp-yang-03 (work in progress), February 2017.

Authors' Addresses

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Adrian Pan
Ericsson

Email: adrian.pan@ericsson.com