Abstract

This document presents a retrieval method YANG Data model for connectionless OAM protocols. It provides technology-independent RPC operations for OAM protocols that use connectionless communication. The retrieval methods model herein presented can be extended to include technology specific details. There are two key benefits of this approach: First, it leads to uniformity between OAM protocols. And second, it support both nested OAM workflows (i.e., performing OAM functions at different or same levels through a unified interface) as well as interactive OAM workflows (i.e., performing OAM functions at same levels through a unified interface).

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://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 May 16, 2018.
1. Introduction

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

1. Monitor network communications (i.e., Reachability Verification, 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 at [RFC7276].

Ping and Traceroute [RFC792] [RFC4443], as well as BFD [RFC5880] are well-known fault verification and isolation tools, respectively, for IP networks. Over the years, different technologies have developed similar toolsets for equivalent purposes.

This document presents an on-demand retrieval method YANG Data model for OAM protocols that use connectionless communication. This model provides technology-independent RPC operations for OAM protocols that use connectionless communication (i.e., connectionless oam). It is separated from the generic YANG model for connectionless OAM [I-D.ietf-lime-yang-connectionless-oam] and can avoid mixing the models for the retrieved-data from the retrieval procedures. It is expected that retrieval procedures would evolve faster than the data model [I-D.ietf-lime-yang-connectionless-oam] and will allow new procedures to be defined for retrieval of the same data defined by the generic YANG data model for connectionless OAM.

2. Conventions used in this document

The following terms are defined in [RFC6241] and are used in this document:

- client
- configuration data
- server
- state data

The following terms are defined in [RFC6020] and are used in this document:

- augment
- data model
- data node

The terminology for describing YANG data models is found in [RFC6020].
2.1. Terminology

TP - Test Point
MAC - Media Access Control
RPC - Remote Procedure Call
RPC Operation - A specific Remote Procedure Call

2.2. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

Each node is printed as:

<status> <flags> <name> <opts> <type>

<status> is one of:
  + for current

<flags> is one of:
  rw for configuration data
  ro for non-configuration data
  -x for rpcs
  -n for notifications

:name> is the name of the node

If the node is augmented into the tree from another module, its name is printed as <prefix>:<name>.

<opts> is one of:
  ? for an optional leaf or choice
  ! for a presence container
  * for a leaf-list or list
  [<keys>] for a list’s keys

<type> is the name of the type for leafs and leaf-lists
3. Overview of the Connectionless OAM retrieval methods Model

This document describes an On-demand retrieval method YANG Data model for OAM protocols that use connectionless communication. This model provides technology-independent retrieval procedures (RPC operations) for connectionless OAM protocols. It provides a flexible way to retrieve the data which defined by the "ietf-connectionless-oam.yang" module [I-D.ietf-lime-yang-connectionless-oam].

3.1. RPC operation definitions

The RPC model facilitates issuing commands to a NETCONF server (in this case to the device that need to execute the OAM command) and obtaining a response.

Under ‘connectionless-oam-methods’ module, we summarize common OAM functions and define two generic RPC operations: ‘continuity-check’ and ‘path-discovery’. In practice, these RPC operations are activated on-demand and supported by corresponding technology-specific OAM tools [RFC7276]. For example, for the IP OAM model, the continuity-check RPC corresponds to the IP Ping [RFC792] [RFC4443], while the path-discovery RPC operation corresponds to IP Traceroute [RFC792] [RFC4443].

Note that the RPC operation presented in this document is the base building block, which is used to derive a model for a technology-specific OAM (i.e., ICMP Ping [RFC792] [RFC4443], and LSP Ping [RFC8029]). This base building block should be extended with corresponding technology specific parameters. To facilitate this for future enhancements to data retrieval methods, the RPCs are captured under a separate module.

The generic ‘tp-address’ grouping is used as data input from different RPCs described in this document. The generic ‘path-discovery-data’ and ‘continuity-check-data’ groupings defined by the "ietf-connectionless-oam.yang" module [I-D.ietf-lime-yang-connectionless-oam] are used as data outputs from different RPCs described in this document. Similar methods including other RPCs can retrieve the data using the same data model (i.e., the "ietf-connectionless-oam.yang" module).

```yaml
rpc continuity-check {
  if-feature cl-oam:continuity-check;
  description "Continuity-check RPC operation as per RFC7276.";
  input {
    uses rpc-input-parameters;
    ...
```

output {
  container response-info {
    leaf protocol-id {
      type identityref {
        base protocol-id;
      }
      mandatory true;
      description
        "Protocol used in CC. ";
    }
    leaf protocol-id-meta-data {
      type identityref {
        base protocol-id-meta-data;
      }
      description
        "An optional meta-data related to the protocol ID.";
    }
    leaf status-code {
      type identityref{
        base status-code;
      }
      mandatory true;
      description
        "Status code for Continuity Check RPC operation.";
    }
    leaf status-sub-code {
      type identityref{
        base status-sub-code;
      }
      mandatory true;
      description
        "Status Sub code for Continuity Check RPC operation.";
    }
    description
      "Status Code and Status Sub Code for continuity check RPC operation.";
    uses cl-oam:continuity-check-data;
  }
}

rpc path-discovery {
  description
    "path discovery RPC operation as per RFC7276.";
  input {
    uses rpc-input-parameters;
    .....
output {
    list response-list {
        key "response-index";
        description "Path discovery response list."
        leaf response-index {
            type uint32;
            mandatory true;
            description "Response index."
        }
        leaf protocol-id {
            type identityref {
                base protocol-id;
            }
            mandatory true;
            description "Protocol used in PD."
        }
        leaf protocol-id-meta-data {
            type identityref {
                base protocol-id-meta-data;
            }
            description "An optional meta-data related to the protocol ID."
        }
        leaf status-code {
            type identityref {
                base status-code;
            }
            mandatory true;
            description "Status code for Path Discovery RPC operation."
        }
        leaf status-sub-code {
            type identityref {
                base status-sub-code;
            }
            mandatory true;
            description "Status Sub code for Path Discovery RPC operation."
        }
    }
}

Snippet of data hierarchy related to RPC operations
3.2. OAM Retrieval Methods Hierarchy

The complete data hierarchy related to the Connectionless OAM Retrieval Methods YANG model is presented below.

module: ietf-connectionless-oam-methods
rpcs:
  ----x continuity-check {cl-oam:continuity-check}?
    |   ---w input
    |     ----w destination-tp
    |     |     ----w tp-location-type identityref
    |     |     ----w mac-address
    |     |     |     ----w mac-address yang:mac-address
    |     |     ----w ipv4-address
    |     |     |     ----w ipv4-address inet:ipv4-address
    |     |     ----w ipv6-address
    |     |     |     ----w ipv6-address inet:ipv6-address
    |     |     ----w tp-attribute
    |     |     |     ----w tp-attribute-type? address-attribute-type
    |     |     |     |     ----w (tp-attribute-value)?
    |     |     |     |     |     ----:(ip-prefix)
    |     |     |     |     |     |     ----w ip-prefix? inet:ip-prefix
    |     |     |     |     |     ----:(bgp)
    |     |     |     |     |     |     ----w bgp? inet:ip-prefix
    |     |     |     |     |     ----:(tunnel)
    |     |     |     |     |     |     ----w tunnel-interface? uint32
    |     |     |     |     |     ----:(pw)
    |     |     |     |     |     |     ----w remote-pe-address? inet:ip-address
    |     |     |     |     |     |     ----w pw-id? uint32
    |     |     |     |     |     ----:(vpls)
    |     |     |     |     |     |     ----w route-distinguisher? rt:route-distinguisher
    |     |     |     |     |     |     ----w sender-ve-id? uint16
    |     |     |     |     |     |     ----w receiver-ve-id? uint16
    |     |     |     |     |     ----:(mpls-mldp)
    |     |     |     |     |     |     ----w (root-address)?
    |     |     |     |     |     |     |     ----:(ip-address)
    |     |     |     |     |     |     |     |     ----w source-address? inet:ip-address
    |     |     |     |     |     |     |     |     ----w group-ip-address? inet:ip-address
    |     |     |     |     |     |     |     |     ----:(vpn)
    |     |     |     |     |     |     |     |     ----w as-number? inet:as-number
    |     |     |     |     |     |     |     |     ----:(global-id)
    |     |     |     |     |     |     |     |     ----w lsp-id? string
    |     |     |     |     |     ----w system-info
    |     |     |     |     |     |     ----w router-id? rt:router-id
    |     |     |     |     |     |     ----w source-interface if:interface-ref
    |     |     |     |     |     |     ----w outbound-interface if:interface-ref
    |     |     |     |     |     |     ----w vrf? cl-oam:routing-instance-ref
    |     |     |     |     |     ----w session-type? enumeration
Internet-Draft     Retrieval Methods CL OAM YANG model     November 2017

| +---w count?          uint32
| +---w ttl?           uint8
| +---w packet-size?    uint32
++-ro output
  +--ro response-info
     | +--ro protocol-id    identityref
     | +--ro protocol-id-meta-data? identityref
     | +--ro status-code     identityref
     | +--ro status-sub-code identityref
  +--ro src-test-point
     | +--ro ni?            routing-instance-ref
     | +--ro tp-location-type identityref
     | +--ro mac-address
        | +--ro mac-address    yang:mac-address
     | +--ro ipv4-address
        | +--ro ipv4-address    inet:ipv4-address
     | +--ro ipv6-address
        | +--ro ipv6-address    inet:ipv6-address
     +--ro tp-attribute
        | +--ro tp-attribute-type? address-attribute-type
        +--ro (tp-attribute-value)?
           +++:(ip-prefix)
              | +--ro ip-prefix?    inet:ip-prefix
           +++:(bgp)
              | +--ro bgp?         inet:ip-prefix
           +++:(tunnel)
              | +--ro tunnel-interface? uint32
           +++:(pw)
              | +--ro remote-pe-address? inet:ip-address
              | +--ro pw-id?       uint32
           +++:(vpls)
              | +--ro route-distinguisher? rt:route-distinguisher
              | +--ro sender-ve-id? uint16
              | +--ro receiver-ve-id? uint16
           +++:(mpls-mldp)
              | +--ro (root-address)?
                 +++:(ip-address)
                    | +--ro source-address?    inet:ip-address
                    | +--ro group-ip-address?    inet:ip-address
                 +++:(vpn)
                    | +--ro as-number?   inet:as-number
                 +++:(global-id)
                    | +--ro lsp-id?     string
     +--ro system-info
        | +--ro router-id?    rt:router-id
        +--ro egress-intf-name? if:interface-ref
     +--ro dest-test-point
        | +--ro ni?         routing-instance-ref
+ro session-delay-statistics

  +ro time-unit-value? identityref
  +ro min-delay-value? uint32
  +ro max-delay-value? uint32
  +ro average-delay-value? uint32

+ro session-jitter-statistics

  +ro unit-value? identityref
  +ro min-jitter-value? uint32
  +ro max-jitter-value? uint32
  +ro average-jitter-value? uint32

---x path-discovery {cl-oam:path-discovery}?

  +w input

    +w destination-tp

      +w tp-location-type identityref
      +w mac-address
      | +w mac-address yang:mac-address
      +w ipv4-address
      | +w ipv4-address inet:ipv4-address
      +w ipv6-address
      | +w ipv6-address inet:ipv6-address
      +w tp-attribute

      +w tp-attribute-type? address-attribute-type
      +w (tp-attribute-value)?

        +:(ip-prefix)
        | +w ip-prefix? inet:ip-prefix
        +:(bgp)
        | +w bgp? inet:ip-prefix
        +:(tunnel)
        | +w tunnel-interface? uint32
        +:(pw)
        | +w remote-pe-address? inet:ip-address
        | +w pw-id? uint32
        +:(vpls)
        | +w route-distinguisher? rt:route-distinguisher
        | +w sender-ve-id? uint16
        | +w receiver-ve-id? uint16
        +:(mpls-mldp)
        | +w (root-address)?

          +:(ip-address)
          | +w source-address? inet:ip-address
          | +w group-ip-address? inet:ip-address
          +:(vpn)
          | +w as-number? inet:as-number
          +:(global-id)
          | +w lsp-id? string

        +w system-info

        +w router-id? rt:router-id

      +w source-interface if:interface-ref
++-ro ni? routing-instance-ref
++-ro tp-location-type identityref
++-ro mac-address
  | +-ro mac-address yang:mac-address
++-ro ipv4-address
  | +-ro ipv4-address inet:ipv4-address
++-ro ipv6-address
  | +-ro ipv6-address inet:ipv6-address
++-ro tp-attribute
  +-ro tp-attribute-type? address-attribute-type
  | +-ro (tp-attribute-value)?
  |   +-:(ip-prefix)
  |     | +-ro ip-prefix? inet:ip-prefix
  |   +-:(bgp)
  |     | +-ro bgp? inet:ip-prefix
  |   +-:(tunnel)
  |     | +-ro tunnel-interface? uint32
  |   +-:(pw)
  |     | +-ro remote-pe-address? inet:ip-address
  |     | +-ro pw-id? uint32
  |   +-:(mpls-mldp)
  |     | +-ro (root-address)?
  |     |   +-:(ip-address)
  |     |     | +-ro source-address? inet:ip-address
  |     |     | +-ro group-ip-address? inet:ip-address
  |     |   +-:(vpn)
  |     |     | +-ro as-number? inet:as-number
  |     |   +-:(global-id)
  |     |     | +-ro lsp-id? string
++-ro system-info
  +-ro router-id? rt:router-id
++-ro sequence-number? uint64
++-ro hop-cnt? uint8
++-ro session-packet-statistics
  | +-ro rx-packet-count? uint32
  | +-ro tx-packet-count? uint32
  | +-ro rx-bad-packet? uint32
  | +-ro tx-packet-failed? uint32
++-ro session-error-statistics
  | +-ro packet-loss-count? uint32
  | +-ro loss-ratio? percentage
  | +-ro packet-reorder-count? uint32
  | +-ro packets-out-of-seq-count? uint32
  | +-ro packets-dup-count? uint32
### ro session-delay-statistics
- **ro time-unit-value?** identityref
- **ro min-delay-value?** uint32
- **ro max-delay-value?** uint32
- **ro average-delay-value?** uint32

### ro session-jitter-statistics
- **ro unit-value?** identityref
- **ro min-jitter-value?** uint32
- **ro max-jitter-value?** uint32
- **ro average-jitter-value?** uint32

### ro session-path-verification-statistics
- **ro verified-count?** uint32
- **ro failed-count?** uint32

### ro path-trace-info-list
- **ro index** uint32
- **ro ni?** routing-instance-ref
- **ro tp-location-type** identityref
- **ro mac-address**
  - **ro mac-address** yang:mac-address
- **ro ipv4-address**
  - **ro ipv4-address** inet:ipv4-address
- **ro ipv6-address**
  - **ro ipv6-address** inet:ipv6-address
- **ro tp-attribute**
  - **ro (tp-attribute-type)?** address-attribute-type
    - **ro (ip-prefix)?**
      - **ro ip-prefix?** inet:ip-prefix
    - **ro (bgp)?**
      - **ro bgp?** inet:ip-prefix
    - **ro (tunnel)?**
      - **ro tunnel-interface?** uint32
    - **ro (pw)?**
      - **ro remote-pe-address?** inet:ip-address
    - **ro (vpls)?**
      - **ro route-distinguisher?** rt:route-distinguisher
    - **ro (mpls-mldp)?**
      - **ro (ip-address)?**
        - **ro source-address?** inet:ip-address
        - **ro group-ip-address?** inet:ip-address
      - **ro (vpn)?**
4. OAM Retrieval Methods YANG Module

<CODE BEGINS> file "ietf-connectionless-oam-methods@2017-09-06.yang"

module ietf-connectionless-oam-methods {
    prefix cloam-methods;
    import ietf-interfaces {
        prefix if;
    }
    import ietf-connectionless-oam {
        prefix cl-oam;
    }
    organization "IETF LIME Working Group";
    contact "Deepak Kumar dekumar@cisco.com
    Qin Wu bill.wu@huawei.com
    S Raghavan srihari@cisco.com
    Zitao Wang wangzitao@huawei.com
    R Rahman rrahman@cisco.com";
    description "This YANG module defines the RPC operations for
    "

connectionless OAM to be used within IETF in a protocol independent manner. It is assumed that each protocol maps corresponding abstracts to its native format. Each protocol may extend the YANG model defined here to include protocol specific extensions; revision 2017-09-06{
  description
  "08 version";
  reference "draft-ietf-lime-yang-connectionless-oam-methods";
}
identity protocol-id {
  description
  "This is base identity for a generic protocol ID. The protocol registry can be found in https://www.iana.org/protocols.";
}
identity protocol-id-internet {
  base protocol-id;
  description
  "Identity for Internet Protocols.";
}
identity protocol-id-proprietary {
  base protocol-id;
  description
  "Identity for proprietary protocol (e.g., IP SLA).";
}
identity protocol-id-sfc {
  base protocol-id;
  description
  "Identity for Service Function Chaining.";
}
identity protocol-id-mpls {
  base protocol-id;
  description
  "MPLS protocol.";
}
identity protocol-id-mpls-tp {
  base protocol-id;
  description
  "MPLS-TP protocol.";
}
identity protocol-id-twamp {
  base protocol-id;
  description
  "TWAMP protocol.";
}
identity protocol-id-bier {
base protocol-id;
description "BIER protocol."
}

identity status-code {
  description "This is Base Identity for status code.";
}

identity success-reach {
  base status-code;
  description "Indicate that the destination being verified is reachable (See RFC7276).";
}

identity fail-reach {
  base status-code;
  description "Indicate that the destination being verified is not reachable (See RFC7276).";
}

identity success-path-verification {
  base status-code;
  description "Indicate that the path verification is performed successfully (See RFC7276).";
}

identity fail-path-verification {
  base status-code;
  description "Indicate that the path verification fails (See RFC7276).";
}

identity status-sub-code {
  description "IdentityBase status sub code.";
}

identity invalid-cc {
  base status-sub-code;
  description "Indicates that the Continuity check message is invalid (See RFC7276).";
}

identity invalid-pd {
  base status-sub-code;
  description "Indicates that the path discovery message is invalid (See RFC7276).";
}

identity protocol-id-meta-data {
  description "This is base identity for meta-data corresponding
to protocol ID.

}  

identity protocol-internet-number {
  base protocol-id-meta-data;
  description
  "Internet Protocol Number for standard
  Internet Protocols (IANA assigned Internet
  Protocol numbers) to help in protocol processing.
  The protocol IDs registry can be found in
  https://www.iana.org/assignments/protocol-numbers
  /protocol-numbers.xhtml."
}

grouping rpc-input-parameters {
  container destination-tp {
    uses cl-oam:tp-address;
    description
    "Destination test point.";
  }
  leaf source-interface {
    type if:interface-ref;
    mandatory true;
    description
    "Source interface.";
  }
  leaf outbound-interface {
    type if:interface-ref;
    mandatory true;
    description
    "Outbound interface.";
  }
  leaf vrf {
    type cl-oam:routing-instance-ref;
    description
    "VRF instance.";
  }
  description
  "Grouping for RPC input parameters";
}

rpc continuity-check {
  if-feature "cl-oam:continuity-check";
  description
  "Continuity-check RPC operation as per RFC7276.";
  input {
    uses rpc-input-parameters;
    uses cl-oam:session-type {
      description
      "If session-type is specified, then session-type
must be set to on-demand;

leaf count {
  type uint32 {
    range 0..4294967295 {
      description
      "The overall number of packet to be transmitted
      by the sender. The value of count will be set
to zero (0) on creation and will thereafter increase
monotonically until it reaches a maximum value of 2^32-1
(4294967295 decimal), when it wraps around and starts
increasing again from zero.";
    }
  }
  default "5";
  description
  "Specifies the number of
  packets that will be sent. By
default, the packet number is
  set to 5.";
}
leaf ttl {
  type uint8;
  default "255";
  description
  "Time to live (TTL) used to limit lifetime
  of data packet transmitted in the network
  and prevent looping. The TTL value is decremented
  for every hop which the packet traverses. If the
  TTL is zero, the data packet will be discarded.";
}
leaf packet-size {
  type uint32 {
    range "64..10000";
  }
  default "64";
  description
  "Packet size of continuity-check message, in octets.
  By default, the packet size is set to 64 octets.";
}
output {
  container response-info {
    leaf protocol-id {
      type identityref {
        base protocol-id;
      }
      mandatory true;
    }
  }
}
description
"Protocol used in continuity check message.
This could be a standard protocol (e.g.,
TCP/IP protocols, MPLS etc.,) or a proprietary
protocol as identified by this field.;
}
leaf protocol-id-meta-data {
  type identityref {
    base protocol-id-meta-data;
  }
  description
  "An optional meta-data related to the protocol ID.
  For e.g., this could be the Internet Protocol number
  for standard Internet Protocols for help in protocol
  processing.;"
}
leaf status-code {
  type identityref{
    base status-code;
  }
  mandatory true;
  description
  "Status code for continuity check RPC operation.
  This could be a basic status code (e.g., destination
  is reachable or destination is not reachable (See RFC7276))
or some customized status code as identified by this field.;"
}
leaf status-sub-code {
  type identityref{
    base status-sub-code;
  }
  mandatory true;
  description
  "An optional status sub code for continuity check
RPC operation. If the basic status code is destination
reachable, this status-sub-code doesn't need to be specified.
If the basic status code is destination unreachable, the
status-sub-code can be used to specify the detailed reasons.
This could be a basic sub-status-code (such as invalid cc) or
other error codes specific to the protocol under use for CC
For example if ICMP is the protocol under use, the error codes
defined in [RFC4443] can be used to specify the reasons specific to
ICMP. These technology specific status-sub-code can be defined
in technology specific models.;"
}

"Status Code and Status sub code for continuity check RPC operation.;"
uses cl-oam:continuity-check-data;
}
}

rpc path-discovery {
  if-feature "cl-oam:path-discovery";
  description
    "Path discovery RPC operation as per RFC7276.";
  input {
    uses rpc-input-parameters;
    uses cl-oam:session-type {
      description
        "If session-type is specified, then session-type
         must be set to on-demand";
    }
    leaf max-ttl {
      type uint8;
      default "255";
      description
        "Maximum TTL indicates the maximum number of hops that
         a packet is permitted to travel before being discarded
         by a router. By default, the maximum TTL is set to 255.";
    }
  }
  output {
    list response-list {
      key "response-index";
      description
        "Path discovery response list.";
      leaf response-index {
        type uint32;
        mandatory true;
        description
          "Response index.";
      }
      leaf protocol-id {
        type identityref {
          base protocol-id;
        }
        mandatory true;
        description
          "Protocol used in PD. This could be a standard
          protocol (e.g., TCP/IP protocols, MPLS etc.,)
          or a proprietary protocol as identified by
          this field.";
      }
      leaf protocol-id-meta-data {
        type identityref {
          base protocol-id-meta-data;
        }
      }
    }
  }
}
base protocol-id-meta-data;
}
description
"An optional meta-data related to the protocol ID. For e.g., this could be the Internet Protocol number for standard Internet Protocols for help in protocol processing.";
}
leaf status-code {
  type identityref{
    base status-code;
  }
  mandatory true;
  description
  "Status code for continuity check RPC operation. This could be a basic status code (e.g., destination is reachable or destination is not reachable) or some customized status code as identified by this field.";
}
leaf status-sub-code {
  type identityref{
    base status-sub-code;
  }
  mandatory true;
  description
  "An optional status sub code for continuity check RPC operation. If the basic status code is destination reachable, this status-sub-code doesn’t need to be specified. If the basic status code is destination unreachable, the status-sub-code can be used to specify the detailed reasons. This could be a basic sub-status-code (such as invalid cc) or other error codes specific to the protocol under use for CC. For example if ICMP is the protocol under use, the error codes defined in [RFC4443] can be used to specify the reasons specific to ICMP. These technology specific status-sub-code can be defined in technology specific models.";
}
}
uses cl-oam:path-discovery-data;
}
}
5. Security Considerations

The YANG module defined in this document 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.

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

- **continuity-check**: Generates continuity check.
- **path-discovery**: Generates path discovery.

These operations are used to retrieve the data from the device that need to execute the OAM command. Unauthorized source access to some sensitive information in the above data may be used for network reconnaissance or lead to Denial-of-Service attack on both the local device and the network.

6. 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:


**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-connectionless-oam-methods

prefix: cloam-methods

reference: RFC XXXX

7. Acknowledgements

The authors of this document would like to thank Elwyn Davies, Alia Atlas, Brian E Carpenter, Greg Mirsky, Adam Roach, Alissa Cooper, Eric Rescorla, Ben Campbell, Benoit Claise, Kathleen Moriarty, Carlos Pignataro, Benjamin Kaduk, and others for their substantive review, comments, and proposals to improve the document.

8. References

8.1. Normative References

[I-D.ietf-lime-yang-connectionless-oam]


8.2. Informative References


Appendix A.  Appendix A.1 Extending Connectionless OAM Method Module Example

The following is an example of extensions possible to "ietf-connectionless-oam-methods" YANG model defined in this document.

The snippet below depicts an example of augmenting the "ietf-connectionless-oam-methods" YANG model with ICMP ping attributes:

```
augment "/cloam-methods:continuity-check" 
  +"/cloam-methods:output"{
    container session-rtt-statistics{
      leaf min-rtt{
        type uint32;
        description
        "This minimum ping round-trip-time (RTT) received.";
      }
      leaf max-rtt{
        type uint32;
        description
        "This maximum ping round-trip-time (RTT) received.";
      }
      leaf avg-rtt{
        type uint32;
        description
        "The current average ping round-trip-time (RTT)";
      }
      description
      "This container presents the ping round-trip-time statistics.";
    }
  }
```

Appendix B.  Appendix A.2 Example of new retrieval procedures Model

As discussed in the Introduction section of this document, the new retrieval procedures can be defined for retrieval of the same data defined by base YANG Data model for connectionless OAM protocols. This appendix demonstrates how the base connectionless OAM data model can be extended to support persistent data retrieval besides on demand retrieval procedures defined in section 3, i.e., first retrieve a persistent-id based on the destination test point location information and then retrieve the export details based on persistent-
id. Internet Protocol Flow Information Export (IPFIX) [RFC7011] or YANG-push [I-D.ietf-netconf-yang-push] are currently outlined here as data export options. Additional export options can be added in future.

The YANG module "example-cl-oam-persistent-methods" shown below is intended as an illustration rather than a real definition of a RPC operation model for persistent data retrieval. For the sake of brevity, this module does not obey all the guidelines specified in [RFC6087].

module example-cl-oam-persistent-methods {
  namespace "http://example.com/cl-oam-persistent-methods";
  import ietf-interfaces {
    prefix if;
  }
  import ietf-connectionless-oam {
    prefix cl-oam;
  }
  import ietf-yang-types {
    prefix yang;
  }
  identity export-method {
    description "Base identity to represent a conceptual export-method.";
  }
  identity ipfix-export {
    base export-method;
    description "IPFIX based export. Configuration provided separately.";
  }
  identity yang-push-export {
    base export-method;
    description "Yang-push from draft-ietf-netconf-yang-push";
  }
  identity protocol-id {
    description "A generic protocol identifier.";
  }
  identity status-code {
    description "Base status code";
  }
  identity success-reach {
    base status-code;
    description
"Indicate that the destination being verified is reachable";
}

identity fail-reach {
  base status-code;
  description
  "Indicate that the destination being verified is not reachable";
}

identity success-path-verification {
  base status-code;
  description
  "Indicate that the path verification is performed successfully.";
}

identity fail-path-verification {
  base status-code;
  description
  "Indicate that the path verification fails.";
}

identity status-sub-code {
  description
  "Base status sub code";
}

identity invalid-cc {
  base status-sub-code;
  description
  "Indicates that the Continuity check message is invalid";
}

identity invalid-pd {
  base status-sub-code;
  description
  "Indicates that the path discovery message is invalid";
}

typedef export-method {
  type identityref {
    base export-method;
  }
  description
    "Export method type.";
}

typedef change-type {
  type enumeration {
    enum "create" {
      description
        "Change due to a create.";
    }
    enum "delete" {
description
  "Change due to a delete.";
}
enum "modify" {
  description
  "Change due to an update.";
}
}
description
  "Different types of changes that may occur.";
}

rpc cc-get-persistent-id {
  if-feature "cl-oam:continuity-check";
  description
  "Obtains continuity-check persistent identification given mapping
  parameters as input.";
  input {
    container destination-tp {
      uses cl-oam:tp-address;
      description
      "Destination test point.";
    }
    uses cl-oam:session-type;
    leaf source-interface {
      type if:interface-ref;
      description
      "Source interface.";
    }
    leaf outbound-interface {
      type if:interface-ref;
      description
      "Outbound interface.";
    }
    leaf vrf {
      type cl-oam:routing-instance-ref;
      description
      "VRF instance.";
    }
  }
  output {
    container error-code {
      leaf protocol-id {
        type identityref {
          base protocol-id;
        }
        mandatory true;
        description
        "Error code.";
      }
    }
  }
}
"Protocol used. This could be a standard protocol (e.g., TCP/IP protocols, MPLS etc.,) or a proprietary protocol as identified by this field."

leaf protocol-id-meta-data {
  type uint64;
  description
    "An optional meta-data related to the protocol ID. For e.g., this could be the Internet Protocol number for standard Internet Protocols for help in protocol processing."
}

leaf status-code {
  type identityref{
    base status-code;
  }
  mandatory true;
  description
    "Status code."
}

leaf status-sub-code {
  type identityref{
    base status-sub-code;
  }
  mandatory true;
  description
    "Sub code for CC."
  description
    "Status code and Sub Code."
}

leaf cc-persistent-id {
  type string;
  description
    "Id to act as a cookie."
}

rpc cc-persistent-get-export-details {
  if-feature "cl-oam:continuity-check";
  description
    "Given the persistent id, gets the configuration options, details related to the configured data export."
  input {

leaf cc-persistent-id {
    type string;
    description
        "Persistent Id for use as a key in search."
}
}
output {
    container error-code {
        leaf protocol-id {
            type identityref {
                base protocol-id;
            }
            mandatory true;
            description
                "Protocol used. This could be a standard
                protocol (e.g., TCP/IP protocols, MPLS etc.,)
                or a proprietary protocol as identified by
                this field."
        }
        leaf protocol-id-meta-data {
            type uint64;
            description
                "An optional meta-data related to the protocol ID.
                For e.g., this could be the Internet Protocol number
                for standard Internet Protocols for help in protocol
                processing."
        }
        leaf status-code {
            type identityref{
                base status-code;
            }
            mandatory true;
            description
                "Status code."
        }
        leaf status-sub-code {
            type identityref{
                base status-sub-code;
            }
            mandatory true;
            description
                "Sub code for CC."
        }
        description
            "Status code and Sub Code."
    }
    leaf data-export-method {
}
type export-method;
  description
    "Type of export in use.";
}

choice cc-trigger {
  description
    "Necessary conditions for periodic or on-change trigger.";
  case periodic {
    description
      "Periodic reports.";
    leaf period {
      type yang:timeticks;
      description
        "Time interval between reports.";
    }
    leaf start-time {
      type yang:date-and-time;
      description
        "Timestamp from which reports were started.";
    }
  }
  case on-change {
    description
      "On-change trigger and not periodic.";
    leaf all-data-on-start {
      type boolean;
      description
        "Full update done on start or not.";
    }
    leaf-list excluded-change {
      type change-type;
      description
        "Changes that will not trigger an update.";
    }
  }
}

rpc pd-get-persistent-id {
  if-feature "cl-oam:path-discovery";
  description
    "Obtains persistent path discovery identification.";
  input {
    container destination-tp {

uses cl-oam:tp-address;
  description
    "Destination test point."
};
uses cl-oam:session-type;
leaf source-interface {
  type if:interface-ref;
  description
    "Source interface.";
}
leaf outbound-interface {
  type if:interface-ref;
  description
    "Outbound interface.";
}
leaf vrf {
  type cl-oam:routing-instance-ref;
  description
    "VRF";
}
}
output {
  list response-list {
    key "response-index";
    description
      "Path discovery response list.";
    leaf response-index {
      type uint32;
      mandatory true;
      description
        "Response index.";
    }
    leaf protocol-id {
      type identityref {
        base protocol-id;
      }
      mandatory true;
      description
        "Protocol used. This could be a standard protocol (e.g., TCP/IP protocols, MPLS etc.,) or a proprietary protocol as identified by this field."
    }
    leaf protocol-id-meta-data {
      type uint64;
      description
        "An optional meta-data related to the protocol ID. For e.g., this could be the Internet Protocol number
for standard Internet Protocols for help in protocol processing.

leaf status-code {
  type identityref {
    base status-code;
  } mandatory true;
  description "Status code for Persistent Path Discovery Information. ";
}

leaf status-sub-code {
  type identityref{
    base status-sub-code;
  } mandatory true;
  description "Sub code for Persistent Path Discovery Information. ";
}

leaf pd-persistent-id {
  type string;
  description "Id to act as a cookie.";
}

rpc pd-persistent-get-export-details {
  if-feature "cl-oam:path-discovery";
  description "Given the persistent id, gets the configuration options, details related to the configured data export.";
  input {
    leaf cc-persistent-id {
      type string;
      description "Persistent ID for use as a key in search.";
    }
  }
  output {
    list response-list {
      key "response-index";
      description "Path discovery response list.";
      leaf response-index {
      }
    }
  }
}
type uint32;
mandatory true;
description
"Response index."

leaf protocol-id {
  type identityref {
    base protocol-id;
  }
  mandatory true;
description
  "Protocol used. This could be a standard
  protocol (e.g., TCP/IP protocols, MPLS etc.,)
  or a proprietary protocol as identified by
  this field."
}

leaf protocol-id-meta-data {
  type uint64;
description
  "An optional meta-data related to the protocol ID.
  For e.g., this could be the Internet Protocol number
  for standard Internet Protocols for help in protocol
  processing."
}

leaf status-code {
  type identityref {
    base status-code;
  }
  mandatory true;
description
  "Status code for Persistent Path Discovery Creation."
}

leaf status-sub-code {
  type identityref {
    base status-sub-code;
  }
  mandatory true;
description
  "Sub code for Persistent Path Discovery Creation."
}

leaf data-export-method {
  type export-method;
description
  "Type of export."
}

choice pd-trigger {
  description
  "Necessary conditions
  to initiate Persistent Path Discovery."
for periodic or on-change trigger.

case periodic {
  description
  "Periodic reports.";
  leaf period {
    type yang:timeticks;
    description
    "Time interval between reports.";
  }
  leaf start-time {
    type yang:date-and-time;
    description
    "Timestamp from which reports are started.";
  }
}

case on-change {
  description
  "On-change trigger and not periodic.";
  leaf all-data-on-start {
    type boolean;
    description
    "Full update done on start or not.";
  }
  leaf-list excluded-change {
    type change-type;
    description
    "Changes that will not trigger an update.";
  }
}

Authors’ Addresses

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

Email: dekumar@cisco.com