Stream: Internet Engineering Task Force (IETF)
RFC: 9617
Category: Standards Track
Published: July 2024
ISSN: 2070-1721
Authors: T. Zhou, Ed. *Huawei*, J. Guichard *Futurewei*, F. Brockners *Cisco Systems*, S. Raghavan *Cisco Systems*

RFC 9617
A YANG Data Model for In Situ Operations, Administration, and Maintenance (IOAM)

Abstract

In situ Operations, Administration, and Maintenance (IOAM) is an example of an on-path hybrid measurement method. IOAM defines a method for producing operational and telemetry information that may be exported using the in-band or out-of-band method. RFCs 9197 and 9326 discuss the data fields and associated data types for IOAM. This document defines a YANG module for the configuration of IOAM functions.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In situ Operations, Administration, and Maintenance (IOAM) is an example of an on-path hybrid measurement method. IOAM defines a method for producing operational and telemetry information that may be exported using the in-band or out-of-band method. The data types and data formats for IOAM data records have been defined in [RFC9197] and [RFC9326]. The IOAM data can be embedded in many protocol encapsulations, such as the Network Service Header (NSH) [RFC9452] and IPv6.

This document defines a data model for the configuration of IOAM capabilities using the YANG data modeling language [RFC7950]. This YANG data model supports five IOAM options, which are as follows:

- Incremental Tracing Option [RFC9197]
- Pre-allocated Tracing Option [RFC9197]
- Direct Export Option [RFC9326]
- Proof of Transit (POT) Option [RFC9197]
- Edge-to-Edge Option [RFC9197]

2. Conventions Used in This Document

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

The following terms are defined in [RFC7950] and are used in this specification:

- augment
- data model
- data node

The terminology for describing YANG data models is found in [RFC7950].

2.1. Tree Diagrams

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

3. Design of the IOAM YANG Data Model

3.1. Overview

The IOAM model is organized as a list of profiles, as shown in the following figure. Each profile associates with one flow and the corresponding IOAM information.

```
module: ietf-ioam
  +--rw ioam
     +--ro info
     |  +--ro timestamp-type?        identityref
     |  +--ro available-interface* [if-name]
     |     +--ro if-name    if:interface-ref
     +--rw admin-config
     |  +--rw enabled?   boolean
     +--rw profiles
        +--rw profile* [profile-name]
           +--rw profile-name                    string
           +--rw filter
           |  +--rw filter-type?   ioam-filter-type
           |  +--rw ace-name?      -> /acl:acls/acl/aces/ace/name
           +--rw protocol-type?                  ioam-protocol-type
           +--rw incremental-tracing-profile {incremental-trace}?
           |  ...
           +--rw preallocated-tracing-profile {preallocated-trace}?
           |  ...
           +--rw direct-export-profile {direct-export}?
           |  ...
           +--rw pot-profile {proof-of-transit}?
           |  ...
           +--rw e2e-profile {edge-to-edge}?
```

The "info" parameter is a container for all the read-only information that assists monitoring systems in the interpretation of the IOAM data.

The "enabled" parameter is an administrative configuration. When it is set to "true", IOAM configuration is enabled for the system. Meanwhile, the IOAM data plane functionality is enabled.

The "filter" parameter is used to identify a flow, where the IOAM profile can apply. There may be multiple filter types. Access Control Lists (ACLs) [RFC8519] provide a common way to specify a flow. Each IOAM profile can associate with an ACE (Access Control Entry). When the matched ACE "forwarding" action is "accept", IOAM actions **MUST** be driven by the accepted packets.

The IOAM data can be encapsulated into multiple protocols, e.g., IPv6 [RFC9486] and the NSH [RFC9452]. The "protocol-type" parameter is used to indicate where IOAM is applied. For example, if "protocol-type" is set to "ipv6", the IOAM ingress node will encapsulate the associated flow with the IPv6-IOAM [RFC9486] format.

In this document, IOAM data includes five encapsulation types, i.e., incremental tracing data, pre-allocated tracing data, direct export data, proof of transit data, and end-to-end data. In practice, multiple IOAM data types can be encapsulated into the same IOAM header. The "profile" parameter contains a set of sub-profiles, each of which relates to one encapsulation type. The configured object may not support all the sub-profiles. The supported sub-profiles are indicated by five defined features, i.e., "incremental-trace", "preallocated-trace", "direct-export", "proof-of-transit", and "edge-to-edge".

This document uses the "ietf-access-control-list" YANG module [RFC8519], the "ietf-interfaces" YANG module [RFC8343], and the "ietf-lime-time-types" YANG module [RFC8532].

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

3.2. Pre-allocated Tracing Profile

To ensure visibility into the entire path that a packet takes within an IOAM domain, the IOAM tracing data is expected to be collected at every node that a packet traverses. The pre-allocated tracing option will create pre-allocated space for each node to populate its information. The "preallocated-tracing-profile" parameter contains the detailed information for the pre-allocated tracing data. This information includes:

node-action: indicates the operation (e.g., encapsulate the IOAM header, transit the IOAM data, or decapsulate the IOAM header) applied to the dedicated flow.

use-namespace: indicates the namespace used for the trace types.

trace-type: indicates the per-hop data to be captured by IOAM-enabled nodes and included in the node data list.

max-length: specifies the maximum length of the node data list in octets. "max-length" is only defined at the encapsulation node.

```
+--rw preallocated-tracing-profile {preallocated-trace}?
   +--rw node-action?            ioam-node-action
   +--rw trace-types
   |  +--rw use-namespace?   ioam-namespace
   |  +--rw trace-type*   ioam-trace-type
   +--rw max-length?             uint32
```

3.3. Incremental Tracing Profile

The incremental tracing option contains a variable node data fields where each node allocates and pushes its node data immediately following the option header. The "incremental-tracing-profile" parameter contains the detailed information for the incremental tracing data. This information is the same as that for the Pre-allocated Tracing Profile; see Section 3.2.

```
+--rw incremental-tracing-profile {incremental-trace}?
   +--rw node-action?            ioam-node-action
   +--rw trace-types
   |  +--rw use-namespace?   ioam-namespace
   |  +--rw trace-type*   ioam-trace-type
   +--rw max-length?             uint32
```

3.4. Direct Export Profile

The direct export option is used as a trigger for IOAM data to be directly exported or locally aggregated without being pushed into in-flight data packets. The "direct-export-profile" parameter contains the detailed information for the direct export data. This information is the same as that for the Pre-allocated Tracing Profile (Section 3.2), but with two more optional variables:

flow-id: used to correlate the exported data of the same flow from multiple nodes and from multiple packets.

enable-sequence-number: indicates whether the sequence number is used in the direct export option.

```
+--rw direct-export-profile {direct-export}?
   +--rw node-action?            ioam-node-action
   +--rw trace-types
   |  +--rw use-namespace?   ioam-namespace
   |  +--rw trace-type*      ioam-trace-type
   +--rw flow-id?             uint32
   +--rw enable-sequence-number? boolean
```

3.5. Proof of Transit Profile

The IOAM proof of transit data is used to support the path or service function chain verification use cases. The "pot-profile" parameter is intended to contain the detailed information for the proof of transit data. The "use-namespace" parameter indicates the namespace used for the POT types. The "pot-type" parameter indicates a particular POT variant that specifies the POT data that is included. There may be several POT types, each having different configuration data. To align with [RFC9197], this document only defines IOAM POT type 0. Users need to augment this module for the configuration of a specific POT type.

```
+--rw pot-profile {proof-of-transit}?
   +--rw use-namespace?   ioam-namespace
   +--rw pot-type?   ioam-pot-type
```

3.6. Edge-to-Edge Profile

The IOAM edge-to-edge option is used to carry data that is added by the IOAM encapsulating node and interpreted by the IOAM decapsulating node. The "e2e-profile" parameter contains the detailed information for the edge-to-edge data. This information includes:

node-action: the same semantic as that provided in Section 3.2.

use-namespace: indicates the namespace used for the edge-to-edge types.

e2e-type: indicates data to be carried from the ingress IOAM node to the egress IOAM node.

```
+--rw e2e-profile {edge-to-edge}?
   +--rw node-action?   ioam-node-action
   +--rw e2e-types
      +--rw use-namespace?   ioam-namespace
      +--rw e2e-type*        ioam-e2e-type
```

4. IOAM YANG Module

The "ietf-ioam" module defined in this document imports typedefs from [RFC8519], [RFC8343], and [RFC8532]. This document also references [RFC9197], [RFC9326], [RFC9486], and [RFC9452].

```
<CODE BEGINS> file "ietf-ioam@2024-07-12.yang"

module ietf-ioam {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-ioam";
  prefix "ioam";

  import ietf-access-control-list {
    prefix "acl";
    reference
      "RFC 8519: YANG Data Model for Network Access Control
       Lists (ACLs)";
  }

  import ietf-interfaces {
    prefix "if";
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }

  import ietf-lime-time-types {
    prefix "lime";
    reference
      "RFC 8532: Generic YANG Data Model for the Management of
       Operations, Administration, and Maintenance (OAM) Protocols
       That Use Connectionless Communications";
  }

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

  contact
    "WG Web:   <https://datatracker.ietf.org/wg/ippm>
     WG List:  <mailto:ippm@ietf.org>
     Editor:   Tianran Zhou
               <mailto:zhoutianran@huawei.com>
     Editor:   Jim Guichard
               <mailto:james.n.guichard@futurewei.com>
     Editor:   Frank Brockners
               <mailto:fbrockne@cisco.com>
     Editor:   Srihari Raghavan
```

```
                <mailto:srihari@cisco.com>";

     description
       "This YANG module specifies a vendor-independent data model
        for In Situ Operations, Administration, and Maintenance
        (IOAM).

        The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
        NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
        'MAY', and 'OPTIONAL' in this document are to be interpreted as
        described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
        they appear in all capitals, as shown here.

        Copyright (c) 2024 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 9617; see the
        RFC itself for full legal notices.";

     revision 2024-07-12 {
       description
         "Initial revision.";
       reference
         "RFC 9617: A YANG Data Model for In Situ Operations,
          Administration, and Maintenance (IOAM)";
     }

    /*
     * FEATURES
     */

     feature incremental-trace
     {
       description
         "This feature indicates that the incremental tracing option
          is supported.";
       reference
         "RFC 9197: Data Fields for In Situ Operations,
          Administration, and Maintenance (IOAM)";
     }

     feature preallocated-trace
     {
       description
         "This feature indicates that the pre-allocated tracing
          option is supported.";
       reference
         "RFC 9197: Data Fields for In Situ Operations,
          Administration, and Maintenance (IOAM)";
     }
```

```
   feature direct-export
   {
     description
       "This feature indicates that the direct export option is
        supported.";
     reference
       "RFC 9326: In Situ Operations, Administration, and
        Maintenance (IOAM) Direct Exporting";
   }

   feature proof-of-transit
   {
     description
       "This feature indicates that the proof of transit option is
        supported.";
     reference
       "RFC 9197: Data Fields for In Situ Operations,
        Administration, and Maintenance (IOAM)";
   }

   feature edge-to-edge
   {
     description
       "This feature indicates that the edge-to-edge option is
        supported.";
     reference
       "RFC 9197: Data Fields for In Situ Operations,
        Administration, and Maintenance (IOAM)";
   }

  /*
   * IDENTITIES
   */
   identity filter {
     description
       "Base identity to represent a filter.  A filter is used to
        specify the flow to apply the IOAM profile.";
   }

   identity acl-filter {
     base filter;
     description
       "Apply Access Control List (ACL) rules to specify the
        flow.";
   }

   identity protocol {
     description
       "Base identity to represent the carrier protocol.  It is
        used to indicate in what layer and protocol the IOAM data
        is embedded.";
   }

   identity ipv6 {
     base protocol;
     description
       "The described IOAM data is embedded in IPv6.";
     reference
```

```
       "RFC 9486: IPv6 Options for In Situ Operations,
        Administration, and Maintenance (IOAM)";
   }

   identity nsh  {
     base protocol;
     description
       "The described IOAM data is embedded in the Network Service
        Header (NSH).";
     reference
       "RFC 9452: Network Service Header (NSH) Encapsulation for
        In Situ OAM (IOAM) Data";
   }

   identity node-action {
     description
       "Base identity to represent the node actions.  It is used to
        indicate what action the node will take.";
   }

   identity action-encapsulate {
     base node-action;
     description
       "This identity indicates that the node is used to
        encapsulate the IOAM packet.";
   }

   identity action-decapsulate {
     base node-action;
     description
       "This identity indicates that the node is used to
        decapsulate the IOAM packet.";
   }

   identity action-transit {
     base node-action;
     description
       "This identity indicates that the node is used to transit
        the IOAM packet.";
   }

   identity trace-type {
     description
       "Base identity to represent trace types.";
   }

   identity trace-hop-lim-node-id {
     base trace-type;
     description
       "This identity indicates the presence of 'Hop_Lim' and
        'node_id' in the node data.";
     reference
       "RFC 9197: Data Fields for In Situ Operations,
        Administration, and Maintenance (IOAM)";
   }

   identity trace-if-id {
     base trace-type;
```

```
      description
        "This identity indicates the presence of 'ingress_if_id' and
         'egress_if_id' (short format) in the node data.";
      reference
        "RFC 9197: Data Fields for In Situ Operations,
         Administration, and Maintenance (IOAM)";
    }

    identity trace-timestamp-seconds {
      base trace-type;
      description
        "This identity indicates the presence of timestamp seconds
         in the node data.";
    }

    identity trace-timestamp-fraction  {
      base trace-type;
      description
        "This identity indicates the presence of a timestamp
         fraction in the node data.";
    }

    identity trace-transit-delay {
      base trace-type;
      description
        "This identity indicates the presence of transit delay in
         the node data.";
    }

    identity trace-namespace-data {
      base trace-type;
      description
        "This identity indicates the presence of namespace-specific
         data (short format) in the node data.";
    }

    identity trace-queue-depth {
      base trace-type;
      description
        "This identity indicates the presence of queue depth in the
         node data.";
    }

    identity trace-checksum-complement {
      base trace-type;
      description
        "This identity indicates the presence of the Checksum
         Complement in the node data.";
      reference
        "RFC 9197: Data Fields for In Situ Operations,
         Administration, and Maintenance (IOAM)";
    }

    identity trace-hop-lim-node-id-wide {
      base trace-type;
      description
        "This identity indicates the presence of 'Hop_Lim' and
         'node_id' (wide format) in the node data.";
```

```
    }

    identity trace-if-id-wide {
      base trace-type;
      description
        "This identity indicates the presence of 'ingress_if_id' and
         'egress_if_id' (wide format) in the node data.";
    }

    identity trace-namespace-data-wide {
      base trace-type;
      description
        "This identity indicates the presence of
         IOAM-namespace-specific data (wide format) in the
         node data.";
    }

    identity trace-buffer-occupancy {
      base trace-type;
      description
        "This identity indicates the presence of buffer occupancy
         in the node data.";
    }

    identity trace-opaque-state-snapshot {
      base trace-type;
      description
        "This identity indicates the presence of the variable-length
         Opaque State Snapshot field.";
    }

    identity pot-type {
      description
        "Base identity to represent Proof of Transit (POT) types.";
    }

    identity pot-type-0 {
      base pot-type;
      description
        "The IOAM field value for the POT type is 0, and POT data is
         a 16-octet field to carry data associated with POT
         procedures.";
    }

    identity e2e-type {
      description
        "Base identity to represent edge-to-edge types.";
    }

    identity e2e-seq-num-64 {
      base e2e-type;
      description
        "This identity indicates the presence of a 64-bit
         sequence number.";
    }

    identity e2e-seq-num-32 {
      base e2e-type;
```

```
     description
       "This identity indicates the presence of a 32-bit
        sequence number.";
   }

   identity e2e-timestamp-seconds {
     base e2e-type;
     description
       "This identity indicates the presence of timestamp seconds
        representing the time at which the packet entered the
        IOAM domain.";
   }

   identity e2e-timestamp-fraction {
     base e2e-type;
     description
       "This identity indicates the presence of a timestamp
        fraction representing the time at which the packet entered
        the IOAM domain.";
   }

   identity namespace {
     description
       "Base identity to represent the Namespace-ID.";
   }

   identity default-namespace {
     base namespace;
     description
       "The Namespace-ID value of 0x0000 is defined as the
        Default-Namespace-ID and MUST be known to all the nodes
        implementing IOAM.";
   }

  /*
   * TYPE DEFINITIONS
   */
   typedef ioam-filter-type {
     type identityref {
       base filter;
     }
     description
       "This type specifies a known type of filter.";
   }

   typedef ioam-protocol-type {
     type identityref {
       base protocol;
     }
     description
       "This type specifies a known type of carrier protocol for
        the IOAM data.";
   }

   typedef ioam-node-action {
     type identityref {
       base node-action;
     }
```

```
      description
        "This type specifies a known type of node action.";
    }

    typedef ioam-trace-type {
      type identityref {
        base trace-type;
      }
      description
        "This type specifies a known trace type.";
    }

    typedef ioam-pot-type {
      type identityref {
        base pot-type;
      }
      description
        "This type specifies a known POT type.";
    }

    typedef ioam-e2e-type {
      type identityref {
        base e2e-type;
      }
      description
        "This type specifies a known edge-to-edge type.";
    }

    typedef ioam-namespace {
      type identityref {
        base namespace;
      }
      description
        "This type specifies the supported namespace.";
    }

   /*
    * GROUP DEFINITIONS
    */

    grouping ioam-filter {
      description
        "A grouping for IOAM filter definitions.";

      leaf filter-type {
        type ioam-filter-type;
        description
          "Filter type.";
      }

      leaf ace-name {
        when "derived-from-or-self(../filter-type, 'ioam:acl-filter')";
        type leafref {
          path "/acl:acls/acl:acl/acl:aces/acl:ace/acl:name";
        }
        description
          "The Access Control Entry name is used to refer to an ACL
           specification.";
```

```
      }
    }

    grouping encap-tracing {
      description
        "A grouping for the generic configuration for the
         tracing profile.";

      container trace-types {
        description
          "This container provides the list of trace types for
           encapsulation.";

        leaf use-namespace {
          type ioam-namespace;
          default default-namespace;
          description
            "This object indicates the namespace used for
             encapsulation.";
        }

        leaf-list trace-type {
          type ioam-trace-type;
          description
            "The trace type is only defined at the encapsulation
             node.";
        }
      }

      leaf max-length {
        when "derived-from-or-self(../node-action,
              'ioam:action-encapsulate')";
        type uint32;
        units bytes;
        description
          "This field specifies the maximum length of the node data
           list in octets.  'max-length' is only defined at the
           encapsulation node.";
      }
    }

    grouping ioam-incremental-tracing-profile {
      description
        "A grouping for the Incremental Tracing Profile.";

      leaf node-action {
        type ioam-node-action;
        default action-transit;
        description
          "This object indicates the action the node needs to
           take, e.g., encapsulation.";
      }

      uses encap-tracing {
        when "derived-from-or-self(node-action,
             'ioam:action-encapsulate')";
      }
    }
```

```
  grouping ioam-preallocated-tracing-profile {
    description
      "A grouping for the Pre-allocated Tracing Profile.";

    leaf node-action {
      type ioam-node-action;
      default action-transit;
      description
        "This object indicates the action the node needs to
         take, e.g., encapsulation.";
    }

    uses encap-tracing {
      when "derived-from-or-self(node-action,
           'ioam:action-encapsulate')";
    }
  }

  grouping ioam-direct-export-profile {
    description
      "A grouping for the Direct Export Profile.";

    leaf node-action {
      type ioam-node-action;
      default action-transit;
      description
        "This object indicates the action the node needs to
         take, e.g., encapsulation.";
    }

    uses encap-tracing {
      when "derived-from-or-self(node-action,
           'ioam:action-encapsulate')";
    }

    leaf flow-id {
      when "derived-from-or-self(../node-action,
           'ioam:action-encapsulate')";
      type uint32;
      description
        "A 32-bit flow identifier.  The field is set at the
         encapsulating node.  The Flow ID can be uniformly
         assigned by a central controller or algorithmically
         generated by the encapsulating node.  The latter approach
         cannot guarantee the uniqueness of the Flow ID, yet the
         probability of conflict is small due to the large Flow ID
         space.  'flow-id' is used to correlate the exported data
         of the same flow from multiple nodes and from multiple
         packets.";
    }

    leaf enable-sequence-number {
      when "derived-from-or-self(../node-action,
           'ioam:action-encapsulate')";
      type boolean;
      default false;
      description
```

```
        "This boolean value indicates whether the sequence number
         is used in the direct export option's 32-bit flow
         identifier.  If this value is set to 'true', the sequence
         number is used.  It is turned off by default.";
    }
  }

  grouping ioam-e2e-profile {
    description
      "A grouping for the Edge-to-Edge Profile.";

    leaf node-action {
      type ioam-node-action;
      default action-transit;
      description
        "This object indicates the action the node needs to
         take, e.g., encapsulation.";
    }

    container e2e-types {
      when "derived-from-or-self(../node-action,
           'ioam:action-encapsulate')";

      description
        "This container provides the list of edge-to-edge types
         for encapsulation.";

      leaf use-namespace {
        type ioam-namespace;
        default default-namespace;
        description
          "This object indicates the namespace used for
           encapsulation.";
      }

      leaf-list e2e-type {
        type ioam-e2e-type;
        description
          "The edge-to-edge type is only defined at the
           encapsulation node.";
      }
    }
  }

  grouping ioam-admin-config {
    description
      "IOAM top-level administrative configuration.";

    leaf enabled {
      type boolean;
      default false;
      description
        "This object is used to control the availability of
         configuration.  It MUST be set to 'true' before anything
         in the /ioam/profiles/profile subtree can be edited.
         If 'false', any configuration in place is not used.";
    }
  }
```

```
/*
 * DATA NODES
 */

 container ioam {
   description
     "IOAM top-level container.";

   container info {
     config false;
     description
       "Describes information, such as units or timestamp format,
        that assists monitoring systems in the interpretation of
        the IOAM data.";

     leaf timestamp-type {
       type identityref {
         base lime:timestamp-type;
       }
       description
         "Type of timestamp, such as Truncated PTP (Precision
          Time Protocol) or NTP.";
     }

     list available-interface {
       key "if-name";
       description
         "A list of available interfaces that support IOAM.";
       leaf if-name {
         type if:interface-ref;
         description
           "This is a reference to the interface name.";
       }
     }
   }

   container admin-config {
     description
       "Contains all the administrative configurations related to
        the IOAM functionalities and all the IOAM profiles.";

     uses ioam-admin-config;
   }

   container profiles {
     description
       "Contains a list of IOAM profiles.";

     list profile {
       key "profile-name";
       description
         "A list of IOAM profiles that are configured on the
          node.  There is no mandatory type of profile (e.g.,
          'incremental-trace', 'preallocated-trace') in the list.
          But at least one profile should be added.";

       leaf profile-name {
```

```
          type string{
            length "1..300";
          }
          description
            "Unique identifier for each IOAM profile.";
        }

        container filter {
          uses ioam-filter;
          description
            "The filter that is used to indicate the flow to apply
             IOAM.";
        }

        leaf protocol-type {
          type ioam-protocol-type;
          description
            "This object is used to indicate the carrier protocol
             where IOAM is applied.";
        }

        container incremental-tracing-profile {
          if-feature incremental-trace;
          presence "Enables the incremental tracing option.";
          description
            "This container describes the profile for the
             incremental tracing option.";

          uses ioam-incremental-tracing-profile;
        }

        container preallocated-tracing-profile {
          if-feature preallocated-trace;
          presence "Enables the pre-allocated tracing option.";
          description
            "This container describes the profile for the
             pre-allocated tracing option.";

          uses ioam-preallocated-tracing-profile;
        }

        container direct-export-profile {
          if-feature direct-export;
          presence "Enables the direct export option.";
          description
            "This container describes the profile for the
             direct export option.";

          uses ioam-direct-export-profile;
        }

        container pot-profile {
          if-feature proof-of-transit;
          presence "Enables the proof of transit (POT) option.";
          description
            "This container describes the profile for the
             POT option.";
```

```
          leaf use-namespace {
            type ioam-namespace;
            default default-namespace;
            description
              "This object indicates the namespace used for the
               POT types.";
          }

          leaf pot-type {
            type ioam-pot-type;
            description
              "The type of a particular POT variant that specifies
               the POT data that is included.";
          }
        }

        container e2e-profile {
          if-feature edge-to-edge;
          presence "Enables the edge-to-edge option.";
          description
            "This container describes the profile for the
             edge-to-edge option.";

          uses ioam-e2e-profile;
        }
      }
    }
  }
}

<CODE ENDS>
```

5. Security Considerations

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

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

There are a number of data nodes defined in 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. These are the subtrees and data nodes and their sensitivity/vulnerability:

/ioam/admin-config:

The items in the container above include the top-level administrative configurations related to the IOAM functionalities and all the IOAM profiles. Unexpected changes to these items could lead to disruption of IOAM functions and/or misbehaving IOAM profiles.

/ioam/profiles/profile: The entries in the list above include the whole IOAM profile configurations. Unexpected changes to these entries could lead to incorrect IOAM behavior for the corresponding flows. Consequently, such changes would impact performance monitoring, data analytics, and the associated reaction to network services.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

/ioam/profiles/profile: The information contained in this subtree might reveal information about the services deployed for customers. For instance, a customer might be given access to monitor the status of their services. In this scenario, the customer's access should be restricted to nodes representing their services so as not to divulge information about the underlying network structure or services.

6. IANA Considerations

IANA has registered the following URI in the "IETF XML Registry" [RFC3688]:

URI: urn:ietf:params:xml:ns:yang:ietf-ioam
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

IANA has registered the following YANG module in the "YANG Module Names" registry [RFC6020]:

Name: ietf-ioam
Namespace: urn:ietf:params:xml:ns:yang:ietf-ioam
Prefix: ioam
Reference: RFC 9617

7. 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, <https://www.rfc-editor.org/info/rfc2119>.

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

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

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

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

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

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

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

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

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

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

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

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

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "YANG Data Model for Network Access Control Lists (ACLs)", RFC 8519, DOI 10.17487/RFC8519, March 2019, <https://www.rfc-editor.org/info/rfc8519>.

[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/info/rfc8532>.

[RFC9197] Brockners, F., Ed., Bhandari, S., Ed., and T. Mizrahi, Ed., "Data Fields for In Situ Operations, Administration, and Maintenance (IOAM)", RFC 9197, DOI 10.17487/RFC9197, May 2022, <https://www.rfc-editor.org/info/rfc9197>.

[RFC9326] Song, H., Gafni, B., Brockners, F., Bhandari, S., and T. Mizrahi, "In Situ Operations, Administration, and Maintenance (IOAM) Direct Exporting", RFC 9326, DOI 10.17487/RFC9326, November 2022, <https://www.rfc-editor.org/info/rfc9326>.

[RFC9452] Brockners, F., Ed. and S. Bhandari, Ed., "Network Service Header (NSH) Encapsulation for In Situ OAM (IOAM) Data", RFC 9452, DOI 10.17487/RFC9452, August 2023, <https://www.rfc-editor.org/info/rfc9452>.

[RFC9486] Bhandari, S., Ed. and F. Brockners, Ed., "IPv6 Options for In Situ Operations, Administration, and Maintenance (IOAM)", RFC 9486, DOI 10.17487/RFC9486, September 2023, <https://www.rfc-editor.org/info/rfc9486>.

Appendix A. An Example of the Incremental Tracing Profile

An example of the Incremental Tracing Profile is depicted in the following figure. This configuration is received by an IOAM ingress node. This node encapsulates the IOAM data in the IPv6 Hop-by-Hop option header. The trace type indicates that each on-path node needs to capture the transit delay and add the data to the IOAM node data list. The incremental tracing data space is variable; however, the node data list must not exceed 512 bytes.

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <ioam xmlns="urn:ietf:params:xml:ns:yang:ietf-ioam">
        <admin-config>
          <enabled>true</enabled>
        </admin-config>
        <profiles>
          <profile>
            <profile-name>ietf-test-profile</profile-name>
            <protocol-type>ipv6</protocol-type>
            <incremental-tracing-profile>
              <node-action>action-encapsulate</node-action>
              <trace-types>
                <use-namespace>default-namespace</use-namespace>
                <trace-type>trace-transit-delay</trace-type>
              </trace-types>
              <max-length>512</max-length>
            </incremental-tracing-profile>
          </profile>
        </profiles>
      </ioam>
    </config>
  </edit-config>
</rpc>
```


Appendix B. An Example of the Pre-allocated Tracing Profile

An example of the Pre-allocated Tracing Profile is depicted in the following figure. This configuration is received by an IOAM ingress node. This node first identifies the target flow by using the ACL parameter "test-acl" and then encapsulates the IOAM data in the NSH. The trace type indicates that each on-path node needs to capture the namespace-specific data in short format and add the data to the IOAM node data list. This node pre-allocates the node data list in the packet with 512 bytes.

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <ioam xmlns="urn:ietf:params:xml:ns:yang:ietf-ioam">
        <admin-config>
          <enabled>true</enabled>
        </admin-config>
        <profiles>
          <profile>
            <profile-name>ietf-test-profile</profile-name>
            <filter>
              <filter-type>acl-filter</filter-type>
              <ace-name>test-acl</ace-name>
            </filter>
            <protocol-type>nsh</protocol-type>
            <preallocated-tracing-profile>
              <node-action>action-encapsulate</node-action>
              <trace-types>
                <use-namespace>default-namespace</use-namespace>
                <trace-type>trace-namespace-data</trace-type>
              </trace-types>
              <max-length>512</max-length>
            </preallocated-tracing-profile>
          </profile>
        </profiles>
      </ioam>
    </config>
  </edit-config>
</rpc>
```

Appendix C. An Example of the Direct Export Profile

An example of the Direct Export Profile is depicted in the following figure. This configuration is received by an IOAM egress node. This node detects the IOAM direct export option in the IPv6 extension header and removes the option to clean all the IOAM data.

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <ioam xmlns="urn:ietf:params:xml:ns:yang:ietf-ioam">
        <admin-config>
          <enabled>true</enabled>
        </admin-config>
        <profiles>
          <profile>
            <profile-name>ietf-test-profile</profile-name>
            <protocol-type>ipv6</protocol-type>
            <direct-export-profile>
              <node-action>action-decapsulate</node-action>
            </direct-export-profile>
          </profile>
        </profiles>
      </ioam>
    </config>
  </edit-config>
</rpc>
```


Appendix D. An Example of the Proof of Transit Profile

The following figure is a simple example of the POT option. This configuration indicates the node to apply POT type 0 with IPv6 encapsulation.


```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <ioam xmlns="urn:ietf:params:xml:ns:yang:ietf-ioam">
        <admin-config>
          <enabled>true</enabled>
        </admin-config>
        <profiles>
          <profile>
            <profile-name>ietf-test-profile</profile-name>
            <protocol-type>ipv6</protocol-type>
            <pot-profile>
              <pot-type>pot-type-0</pot-type>
            </pot-profile>
          </profile>
        </profiles>
      </ioam>
    </config>
  </edit-config>
</rpc>
```

Appendix E. An Example of the Edge-to-Edge Profile

The following figure shows an example of the edge-to-edge option. This configuration is received by an IOAM egress node. This node detects the IOAM edge-to-edge option in the IPv6 extension header and removes the option to clean all the IOAM data. As the IOAM egress node, it may collect the edge-to-edge data and deliver it to the data-exporting process.

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <ioam xmlns="urn:ietf:params:xml:ns:yang:ietf-ioam">
        <admin-config>
          <enabled>true</enabled>
        </admin-config>
        <profiles>
          <profile>
            <profile-name>ietf-test-profile</profile-name>
            <protocol-type>ipv6</protocol-type>
            <e2e-profile>
              <node-action>action-decapsulate</node-action>
            </e2e-profile>
          </profile>
        </profiles>
      </ioam>
    </config>
  </edit-config>
</rpc>
```

Acknowledgements

For their valuable comments, discussions, and feedback, we wish to acknowledge Greg Mirsky, Reshad Rahman, Tom Petch, Mickey Spiegel, Thomas Graf, Alex Huang Feng, and Justin Iurman.

Authors' Addresses

Tianran Zhou (EDITOR)
Huawei
156 Beiqing Rd.
Beijing
100095
China
Email: zhoutianran@huawei.com

Jim Guichard
Futurewei
United States of America
Email: james.n.guichard@futurewei.com

Frank Brockners
Cisco Systems
Hansaallee 249, 3rd Floor
40549 Düsseldorf, Nordrhein-Westfalen
Germany
Email: fbrockne@cisco.com

Srihari Raghavan
Cisco Systems
Tril Infopark Sez, Ramanujan IT City
Neville Block, 2nd floor, Old Mahabalipuram Road
Chennai 600113
Tamil Nadu
India
Email: srihari@cisco.com