Abstract

SR P2MP policies are set of policies that enable architecture for P2MP service delivery.

This document defines a YANG data model for SR P2MP Policy Configuration and operation.

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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." The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html
1. Introduction

This document defines a YANG data model for P2MP SR Policy configuration and operation.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].
3. Design of the Data Model

submodule: router-p2mp-segment (belongs-to root)
  +++rw p2mp-segment!
    +++rw admin-state? boolean
    +++rw p2mp-policy* [p2mp-policy-name]
      ++-rw p2mp-policy-name string
      ++-rw root-address? inet:ipv4-address
      ++-rw tree-id? uint32
      ++-rw admin-state? boolean
      ++-ro oper-state? boolean
      +++rw leaf-list* [leaf-address]
        +++rw leaf-address inet:ipv4-address
        +++rw admin-state? boolean
      +++rw replication-policy* [replication-policy-name]
        ++-rw replication-policy-name string
        ++-rw root-address? inet:ipv4-address
        ++-rw tree-id? uint32
        ++-rw node-address? inet:ipv4-address
        ++-rw admin-state? boolean
        ++-ro oper-state? boolean
      +++rw candidate-path* [candidate-path-name]
        +++rw candidate-path-name string
        +++rw origin? enumeration
        +++rw originator-asn? uint32
        +++rw originator-node-address? inet:ipv4-address
        +++-rw descriminator? uint32
        +++-rw admin-state? boolean
        +++-ro oper-state? boolean
        +++-rw plsp-id? uint32
        +++-rw preference? uint32
        +++-rw incoming-replication-sid? uint32
        +++-rw operation? enumeration
      +++rw next-hop-id* [next-hop-id]
        ++-rw next-hop-id uint32
        ++-rw next-hop-address? inet:ipv4-address
        ++-rw next-hop-interface-name? if:interface-ref
        ++-rw protect-nexthop-id? uint32
        ++-rw weight? uint32
        +++rw (label)?
          +++:(outgoing-replication-sid)
            ++-rw out-replication-sid* [index]
            ++-rw index uint32
          +++:(outgoing-sr-policy-sid-list)
            ++-rw sr-policy-sid-list* [index]
            ++-rw index uint32
          +++:(outgoing-sr-policy)
            +++rw sr-policy? string
4. Configuration

This Module augments the "/rt:routing:" with a treeSID container. This container defines all the configuration parameter related to treeSID and P2MP SR Policy for this particular routing. The P2MP SR policy contains replication policy which in order contain candidate-path and the next-hop-groups for each OIF in the replication Policy. It should be noted that two disjoint replication policies can be connected via a SR Policy as per draft-ietf-spring-segment-routing-policy.

5. Control plane configuration

6. States

7. Yang Data Model
<CODE BEGINS> file "ietf-p2mp-policy@2019-07-04.yang"
module ietf-tree-sid {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:p2mp-policy-segment";
  // replace with IANA namespace when assigned
  prefix tree-sid;

  import ietf-inet-types {
    prefix "inet";
  }

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-routing-types {
    prefix "rt-types";
  }

  import ietf-routing {
    prefix "rt";
  }

  import ietf-interfaces {
    prefix "if";
  }

  import ietf-ip {
    prefix ip;
  }

  organization
    "IETF SPRING Working Group";

  contact
    "WG Web:  <http://tools.ietf.org/wg/spring/>"
    "WG List:  <mailto:spring@ietf.org>"

    "WG Chair: Bruno Decraene"
    "<mailto:bruno.decreaene@orange.com>"

    "WG Chair: Rob Shakir"
    "<mailto:robs@google.com>"

    "Editor: Hooman Bidgoli"
    "<mailto:hooman.bidgoli@nokia.com>"

    "Editor: Tanmoy Kundu"
    "<mailto:tanmoy.kundu@nokia.com>"
The module defines a collection of YANG definitions for p2mp policy module.

```
revision 2019-07-04 {
  description
    "First draft.";
  reference
    "RFC XXXX: A YANG Data Model for TREE-SID";
}

submodule router-p2mp-segment {
  belongs-to root                  { prefix "root"; }
  import ietf-inet-types          { prefix "inet"; }
  import ietf-interfaces          { prefix "if"; }
  container p2mp-segment {
    presence "Configure Tree SID parameters.";
    leaf admin-state {
      type boolean;
      default false;
      description
        "Administratively enable/disable Tree SID.";
    }
    list p2mp-policy {
      key "p2mp-policy-name";
      uses p2mp-policy-key;
      leaf root-address {
        type inet:ipv4-address;
        description
          "Root address of the tree.";
      }
      leaf tree-id {
        type uint32;
        description
          "Tree ID uniquely identifies a tunnel in the root,
           this also represents a specific constraint. Also known as
           color and/or p2mp-id";
      }
      leaf admin-state {
        type boolean;
        default false;
      }
    }
```
description
"Administratively enable/disable Tree SID p2mp policy.";
}
leaf oper-state {
  type boolean;
  default false;
  config false;
  description
  "Tree SID p2mp policy operational state based on users.";
}
list leaf-list {
  key "leaf-address";
  uses leaf-list-key;
  leaf admin-state {
    type boolean;
    default false;
    description
    "Administratively enable/disable Tree SID p2mp policy.";
  }
}
list replication-policy {
  key "replication-policy-name";
  uses replication-policy-key;
  leaf root-address {
    type inet:ipv4-address;
    description
    "Root address of the tree.";
  }
  leaf tree-id {
    type uint32;
    description
    "Tree ID uniquely indentifies a tunnel in the root,
    this also represent a specific constraint. Also known as
    color and/or p2mp-id";
  }
  leaf node-address {
    type inet:ipv4-address;
    description
    "Node address for which this replication policy is used.";
  }
  leaf admin-state {
    type boolean;
    default false;
description
    "Administratively enable/disable Tree SID replication policy.";
}
leaf oper-state {
    type boolean;
    default false;
    config false;
    description
    "Tree SID replication policy operational state based on users.";
}
list candidate-path {
    description
    "Candidate path is Tree SID or SID list representing a unique path from root to a specific endpoint using the tree-id constraint.";
    key "candidate-path-name";
    uses candidate-path-key;
    leaf admin-state {
        type boolean;
        default false;
        description
        "Administratively enable/disable Tree SID candidate path.";
    }
    leaf oper-state {
        type boolean;
        default false;
        config false;
        description
        "candidate-path operational state based on ilm programming.";
    }
    leaf plsp-id {
        type uint32;
        default 0;
        description
        "PLSP-ID is an unique identifier assigned by controller and remain unchanged throughout the life of a LSP(candidate path).";
    }
    leaf preference {
        type uint32;
        default 100;
        description
        "Preference determines the best preferred candidate-path among list of candidate path towards a leaf. Higher preference is choosen.";
    }
}
leaf incoming-replication-sid{
    type uint32;
    default 0;
    description
        "The incoming label for transit and leaf routers, root it
         is 0";
}
leaf operation {
    type enumeration {
        enum push { value 0; }
        enum pop  { value 1; }
        enum swap { value 2; }
    }
    default push;
    description
        "Label operation";
}

list next-hop-id {
    description
        "Identifies each nexthop in a candidate path.";
    key "next-hop-id";
    uses next-hop-id-key;
    leaf next-hop-address {
        type inet:ipv4-address;
        default 0.0.0.0;
        description
            "Nexthop address of the desitnation.";
    }
    leaf next-hop-interface-name {
        type if:interface-ref;
        description
            "Next hop out going interface.";
    }
    leaf protect-nexthop-id {
        type uint32;
        description
            "Nexthop protection id.";
    }
    leaf weight {
        type uint32;
        description
            "weight for weighted load balancing";
    }
}

choice label {
list outgoing-replication-sid {
    key index;
    uses sid-list-key;
    description "Outgoing replication label for this nexthop.";
}
list outgoing-sr-policy-sid-list {
    key index;
    uses sr-policy-sid-key;
    description "Outgoing sr-policy sid for this nexthop.";
}
leaf outgoing-sr-policy {
    type string;
    description "SR policy name to be referenced";
}

// -------------------------- GROUPINGS ---------------------------
grouping p2mp-policy-key {
    leaf p2mp-policy-name {
        type string;
        description "P2MP policy name to be referenced by mvpn pmsi.";
    }
}
grouping replication-policy-key {
    leaf replication-policy-name {
        type string;
        description "Replication policy name to be referenced by mvpn pmsi.";
    }
}
grouping leaf-list-key {
    leaf leaf-address{
        type inet:ipv4-address;
        description "leaf address of the this p2mp-tree";
    }
}
grouping candidate-path-key {

gleaf candidate-path-name {
    type string;
    description
    "A candidate path name equivalent to a LSP.";
}

gleaf origin {
    type enumeration {
        enum pcep { value 10; }
        enum bgp-sr-policy { value 20; }
        enum configuration { value 30; }
    }
    description
    "Protocol-Origin of a candidate path is an 8-bit value which identifies the component or protocol that originates or signals the candidate path.";
}

gleaf originator-asn {
    type uint32;
    description
    "represented as a 4 byte number";
}

gleaf originator-node-address {
    type inet:ipv4-address;
    description
    "128 bit value, IPv4 address are encoded in lower 32 bit.";
}

gleaf descriminator {
    type uint32;
    description
    "The Discriminator is a 32 bit value associated with a candidate path that uniquely identifies it within the context of an SR Policy from a specific Protocol-Origin";
}
}

grouping next-hop-id-key {

gleaf next-hop-id {
    type uint32;
    description
    "Nexthop group index";
}
}
6. IANA Considerations

   This document contains no actions for IANA.

7. Security Considerations

   TBD

8. References

8.1. Normative References

8.2. Informative References


7. Acknowledgments

Authors’ Addresses