TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2021

T. Saad
V. Beeram
Juniper Networks
B. Wen
Comcast
D. Ceccarelli
Ericsson
S. Peng
R. Chen
ZTE Corporation
LM. Contreras
Telefonica
X. Liu
Volta Networks

February 22, 2021

YANG Data Model for Slice Policy
draft-bestbar-teas-yang-slice-policy-00

Abstract

A slice policy is a policy construct that enables instantiation of mechanisms in support of IETF network slice specific control and data plane behaviors on select topological elements. This document defines a YANG data model for the management of slice policies on slice policy capable nodes and controllers in IP/MPLS networks.

Requirements Language

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

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 August 26, 2021.

Table of Contents

1. Introduction

An IETF network slice [I-D.ietf-teas-ietf-network-slice-definition] is a well-defined structure of connectivity requirements and associated network behaviors. An IETF Network Slice Controller (NSC) can realize an IETF network slice by mapping it to a slice aggregate [I-D.bestbar-teas-ns-packet]. A slice aggregate comprises of one or more IETF network slice traffic streams. The NSC uses a policy construct called the slice policy to enable the instantiation of mechanisms in support of IETF network slice specific control and data plane behaviors on select topological elements. The enforcement of the slice policy results in the creation of a slice aggregate.

A slice policy specifies the topology associated with the slice aggregate and dictates how a slice aggregate can be realized in IP/MPLS networks using one of three modes. The slice policy dictates if the partitioning of the shared network resources can be achieved in (a) just the data plane or in (b) just the control plane or in (c) both the control and data planes.

The slice policy modes (a) and (c) require the forwarding engine on each slice policy capable node to identify the traffic belonging to a specific slice aggregate and to apply the corresponding Per-Hop Behavior (PHB) that determines the forwarding treatment of the packets belonging to the slice aggregate. The identification of the slice aggregate that the packet belongs to and the corresponding forwarding treatment that needs to be applied to the packet is dictated by the slice policy.

The slice policy modes (b) and (c) require the distributed/centralized resource reservation manager in the control plane to manage slice aggregate resource reservation. The provisions for enabling slice aggregate aware traffic engineering are dictated by the slice policy.

This document defines a YANG data model for the management of slice policies on slice policy capable nodes and controllers in IP/MPLS networks.

1.1. Terminology

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

The reader is expected to be familiar with the terminology specified in [I-D.ietf-teas-ietf-network-slice-definition], [I-D.nsdt-teas-ns-framework] and [I-D.bestbar-teas-ns-packet]. The

term "Network Slice" used in this document must be interpreted as "IETF Network Slice" [I-D.ietf-teas-ietf-network-slice-definition].

1.2. Tree Structure

A simplified graphical representation of the data model is presented in Appendix A of this document. The tree format defined in [RFC8340] is used for the YANG data model tree representation.

2. Slice Policy Data Model

2.1. Model Usage

The onus is on the IETF network slice controller to consume the service layer network slice intent and realize it with an appropriate slice policy. Multiple IETF network slices can be mapped to the same slice aggregate resulting in the application of the same slice policy. The network wide consistent slice policy definition (provided by the data model defined in this document) is distributed to the slice policy capable nodes and controllers as shown in Figure 1. The specification of the network slice intent on the northbound interface of the controller and the mechanism used to associate the network slice to a slice policy are outside the scope of this document.

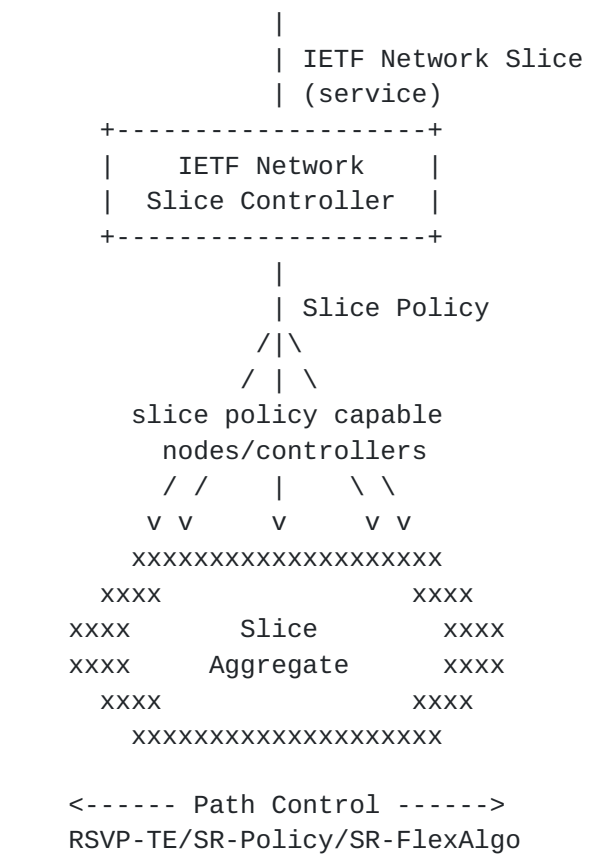

Figure 1: Slice Policy Instantiation

2.2. Model Structure

The high-level model structure defined by this document is as shown below:

```
module: ietf-slice-policy
  +--rw network-slicing!
     +--rw phbs
     |  +--rw phb* [id]
     |  .............
     +--rw topology-filters
     |  +--rw topology-filter* [name]
     |     .............
     +--rw slice-policies
        +--rw slice-policy* [name]
           +  .............
           +--rw resource-reservation
           |  .............
           +--rw slice-selectors
           |  +--rw slice-selector* [index]
           |     .............
           +--rw phb?                    slice-policy-phb-ref
           +--rw member-topologies
              +--rw member-topology* [topology-filter]
                 .............
```

In addition to the set of slice policies, the top-level container also includes placeholders for the set of PHBs and the set of topology filters that are referenced by the slice policies.

2.3. Per-Hop-Behaviors

The 'phbs' container carries a list of PHB entries. Each of these entries can be referenced by one or more slice policies. A PHB entry can either carry a reference to a generic PHB profile available on the node or carry a custom PHB profile. The custom PHB profile includes attributes to construct a slice aggregate specific QoS profile and any classes within it.

```
     +--rw phbs
     |  +--rw phb* [id]
     |     +--rw id                                  uint16
     |     +--rw (profile-type)?
     |        +--:(profile)
     |        |  +--rw profile?                      string
     |        +--:(custom-profile)
     |           .............
```

2.4. Topology Filters

The 'topology-filters' container carries a list of topology filters. Each topology filter entry could either reference a predefined

topology or specify the rules to construct a customized topology using a set of include-any, include-all and exclude filters.

```
     +--rw topology-filters
     |  +--rw topology-filter* [name]
     |     +--rw name                                  string
     |     +--rw (topology-filter-type)?
     |        +--:(standard-topology)
     |        |  +--rw (standard-topo-type)?
     |        |     +--:(flex-algo)
     |        |     |  +--rw algo-id?                 uint8
     |        |     |  +--rw mt-id?                   uint16
     |        |     +--:(te-topo)
     |        |        +--rw te-topology-identifier
     |        |           +--rw provider-id?   te-global-id
     |        |           +--rw client-id?     te-global-id
     |        |           +--rw topology-id?   te-topology-id
     |        +--:(custom-topology)
     |           +--rw include-any
     |           |  +--rw link-affinity*   string
     |           |  +--rw link-name*       string
     |           |  +--rw node-prefix*     inet:ip-prefix
     |           |  +--rw as*              inet:as-number
     |           +--rw include-all
     |           |  +--rw link-affinity*   string
     |           |  +--rw link-name*       string
     |           |  +--rw node-prefix*     inet:ip-prefix
     |           |  +--rw as*              inet:as-number
     |           +--rw exclude
     |              +--rw link-affinity*   string
     |              +--rw link-name*       string
     |              +--rw node-prefix*     inet:ip-prefix
     |              +--rw as*              inet:as-number
```

2.5. Slice Policies

The 'slice-policies' container carries a list of slice policies. Each slice-policy entry is identified by a name and holds the set of attributes needed to instantiate a slice aggregate. The four key elements of each slice-policy entry are discussed in the following sub-sections.

2.5.1. Resource Reservation

The 'resource-reservation' container carries data nodes that are used to support slice aggregate aware bandwidth engineering. The data nodes in this container facilitate preference-based preemption of slice aggregate aware TE paths, sharing of resources amongst a group

of slice aggregates and backup slice aggregate path bandwidth protection.

```
           +--rw resource-reservation
           |  +--rw preference?                        uint16
           |  +--rw (max-bw-type)?
           |  |  +--:(bw-value)
           |  |  |  +--rw maximum-bandwidth?           uint64
           |  |  +--:(bw-percentage)
           |  |     +--rw maximum-bandwidth-percent?
           |  |             rt-types:percentage
           |  +--rw shared-resource-groups*            uint32
           |  +--rw protection
           |     +--rw backup-sa-id?                     uint32
           |     +--rw (backup-bw-type)?
           |        +--:(backup-bw-value)
           |        |  +--rw backup-bandwidth?           uint64
           |        +--:(backup-bw-percentage)
           |           +--rw backup-bandwidth-percent?
           |                   rt-types:percentage
```

2.5.2. Slice Selectors

The 'slice-selectors' container carries a set of data plane field selectors which are used to identify the packets belonging to the given slice aggregate. Each slice-selector entry in the list has an index associated with it. The slice selector with the lowest index is the default slice selector used by all the topological elements that are members of the given slice policy. The other entries are used only when there is a need to override the default slice selector on some select topological elements.

```
            +--rw slice-selectors
            |  +--rw slice-selector* [index]
            |     +--rw index      uint16
            |     +--rw mpls
            |     |  +--rw (ss-mpls-type)?
            |     |     +--:(label-value)
            |     |     |  +--rw label?
            |     |     |  |       rt-types:mpls-label
            |     |     |  +--rw label-position?          identityref
            |     |     |  +--rw label-position-offset?   uint8
            |     |     +--:(label-ranges)
            |     |        +--rw label-range* [index]
            |     |           +--rw index                    string
            |     |           +--rw start-label?
            |     |           |       rt-types:mpls-label
            |     |           +--rw end-label?
            |     |           |       rt-types:mpls-label
            |     |           +--rw label-position?
            |     |           |       identityref
            |     |           +--rw label-position-offset?   uint8
            |     +--rw ipv4
            |     |  +--rw destination-prefix*   inet:ipv4-prefix
            |     +--rw ipv6
            |     |  +--rw (ss-ipv6-type)?
            |     |     +--:(ipv6-destination)
            |     |     |  +--rw destination-prefix*
            |     |     |          inet:ipv6-prefix
            |     |     +--:(ipv6-flow-label)
            |     |        +--rw slid-flow-labels
            |     |           +--rw slid-flow-label* [slid]
            |     |              +--rw slid       inet:ipv6-flow-label
            |     |              +--rw bitmask?   uint32
            |     +--rw acl-ref*   slice-policy-acl-ref
```

2.5.3. Per-Hop-Behavior

The 'phb' leaf carries a reference to the appropriate PHB that needs to be applied for the given slice aggregate. Unless specified otherwise, this is the default phb to be used by all the topological elements that are members of the given slice policy.

```
            +--rw phb?                    slice-policy-phb-ref
```

2.5.4. Member Topologies

The 'member-topologies' container consists of a set of member topologies. Each member topology references a topology filter. The topological elements that satisfy the membership criteria can

optionally override the default PHB and/or the default slice selector.

```
           +--rw member-topologies
              +--rw member-topology* [topology-filter]
                 +--rw topology-filter
                 |       slice-policy-topo-filter-ref
                 +--rw slice-selector-override?   slice-policy-ss-ref
                 +--rw phb-override?
                         slice-policy-phb-ref
```

2.6. YANG Module

```
<CODE BEGINS> file "ietf-slice-policy@2021-02-22.yang"
module ietf-slice-policy {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-slice-policy";
  prefix "sl-pol";

  import ietf-inet-types {
    prefix "inet";
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference
      "RFC 8294: Common YANG Data Types for the Routing Area";
  }

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

  import ietf-te-types {
    prefix te-types;
    reference
      "RFC 8776: Common YANG Data Types for Traffic Engineering";
  }

  organization
    "IETF Traffic Engineering Architecture and Signaling (TEAS)
     Working Group.";
```

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

        Editor:   Vishnu Pavan Beeram
                  <mailto:vbeeram@juniper.net>

        Editor:   Tarek Saad
                  <mailto:tsaad@juniper.net>

        Editor:   Bin Wen
                  <mailto:Bin_Wen@cable.comcast.com>

        Editor:   Daniele Ceccarelli
                  <mailto:daniele.ceccarelli@ericsson.com>

        Editor:   Shaofu Peng
                  <mailto:peng.shaofu@zte.com.cn>

        Editor:   Ran Chen
                  <mailto:chen.ran@zte.com.cn>

        Editor:   Luis M. Contreras
                  <mailto:luismiguel.contrerasmurillo@telefonica.com>

        Editor:   Xufeng Liu
                  <mailto:xufeng.liu.ietf@gmail.com>";

     description
       "This YANG module defines a data model for managing slice
        policies on slice policy capable nodes and controllers.

        Copyright (c) 2021 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 Simplified BSD License set
        forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX; see the
        RFC itself for full legal notices.";

     revision "2021-02-22" {
       description "Initial revision.";
       reference
```

```
        "RFC XXXX: YANG Data Model for Slice Policies.";
    }


    /*
     * I D E N T I T I E S
     */


    /*
     * Identity - MPLS Slice Selector Label Position Type
     */

    identity ss-mpls-label-position-type {
      description
        "Base identity for the position of the MPLS label that is used
         for slice selection.";
    }

    identity ss-mpls-label-position-top {
      base ss-mpls-label-position-type;
      description
        "MPLS label that is used for slice selection is at the top of
         the label stack.";
    }

    identity ss-mpls-label-position-bottom {
      base ss-mpls-label-position-type;
      description
        "MPLS label that is used for slice selection is either at the
         bottom or at a specific offset from the bottom of the label
         stack.";
    }

    identity ss-mpls-label-position-indicator {
      base ss-mpls-label-position-type;
      description
        "MPLS label that is used for slice selection is preceded by
         a special purpose indicator label in the label stack.";
    }

    /*
     * Identity - S-PHB Class Direction
     */

    identity s-phb-class-direction {
      description
        "Base identity for the direction of traffic to which the Slice
```

```
          PHB class profile is applied.";
      }

      identity s-phb-class-direction-in {
        base s-phb-class-direction;
        description
          "Slice PHB class profile is applied to incoming traffic.";
      }

      identity s-phb-class-direction-out {
        base s-phb-class-direction;
        description
          "Slice PHB class profile is applied to outgoing traffic.";
      }

      identity s-phb-class-direction-in-out {
        base s-phb-class-direction;
        description
          "Slice PHB class profile is applied to both incoming and
           outgoing directions of traffic.";
      }

      /*
       * Identity - S-PHB Class Priority
       */

      identity s-phb-class-priority {
        description
          "Base identity for the priority of the child class scheduler.";
      }

      identity s-phb-class-priority-low {
        base s-phb-class-priority;
        description
          "Priority of the child class scheduler is low.";
      }

      identity s-phb-class-priority-strict-high {
        base s-phb-class-priority;
        description
          "Priority of the child class scheduler is strict-high.";
      }

      /*
       * Identity - S-PHB Class Drop Probability
       */

      identity s-phb-class-drop-probability {
```

```
    description
      "Base identity for the drop probability applied to packets
       exceeding the CIR of the class queue.";
  }

  identity s-phb-class-drop-probability-low {
    base s-phb-class-drop-probability;
    description
      "Low drop probability applied to packets exceeding the CIR of
       the class queue.";
  }

  identity s-phb-class-drop-probability-medium {
    base s-phb-class-drop-probability;
    description
      "Medium drop probability applied to packets exceeding the CIR
       of the class queue.";
  }

  identity s-phb-class-drop-probability-high {
    base s-phb-class-drop-probability;
    description
      "High drop probability applied to packets exceeding the CIR of
       the class queue.";
  }

  /*
   * T Y P E D E F S
   */

  typedef slice-policy-acl-ref {
    type leafref {
      path "/acl:acls/acl:acl/acl:name";
    }
    description
      "This type is used to reference an ACL.";
  }

  typedef slice-policy-ss-ref {
    type leafref {
      path "/network-slicing/slice-policies/slice-policy/"
        + "slice-selectors/slice-selector/index";
    }
    description
      "This type is used to reference a Slice Selector (SS).";
  }

  typedef slice-policy-phb-ref {
```

```
      type leafref {
        path "/network-slicing/phbs/phb/"
          + "id";
      }
      description
        "This type is used to reference a Slice Policy Per-Hop
         Behavior (S-PHB).";
    }

    typedef slice-policy-topo-filter-ref {
      type leafref {
        path "/network-slicing/topology-filters/topology-filter/"
          + "name";
      }
      description
        "This type is used to reference a Slice Policy Topology.";
    }

    /*
     * G R O U P I N G S
     */

    /*
     * Grouping - Slice Selector MPLS: Label location specific fields
     */
    grouping sl-pol-ss-mpls-label-location {
      description
        "Grouping for MPLS (SS) label location specific fields.";
      leaf label-position {
        type identityref {
          base ss-mpls-label-position-type;
        }
        description
          "MPLS label position - top, bottom with offset, Slice label
           indicator.";
      }
      leaf label-position-offset {
        when "derived-from-or-self(../label-position,"
          + "'sl-pol:ss-mpls-label-position-bottom')" {
          description
            "MPLS label position offset is relevant only when the
             label-position is set to 'bottom'.";
        }
        type uint8;
        description
          "MPLS label position offset.";
      }
    }
```

```
    /*
     * Grouping - Slice Selector (SS)
     */
    grouping sl-pol-slice-selector {
      description
        "Grouping for Slice Selectors.";
      container slice-selectors {
        description
          "Container for Slice Selectors.";
        list slice-selector {
          key "index";
          description
            "List of Slice Selectors - this includes the default
             selector and others that are used for overriding the
             default.";
          leaf index {
            type uint16;
            description
              "An index to identify an entry in the slice-selector
               list. The entry with the lowest index is the
               default slice-selector.";
          }
          container mpls {
            description
              "Container for MPLS Slice Selector.";
            choice ss-mpls-type {
              description
                "Choices for MPLS Slice Selector.";
              case label-value {
                leaf label {
                  type rt-types:mpls-label;
                  description
                    "MPLS Slice Selector Label is explicitly
                     specified.";
                }
                uses sl-pol-ss-mpls-label-location;
              }
              case label-ranges {
                list label-range {
                  key "index";
                  unique "start-label end-label";
                  description
                    "MPLS Slice Selector Label is picked from a
                     specified set of label ranges.";
                  leaf index {
                    type string;
                    description
                      "A string that uniquely identifies a label
```

```
                    range.";
                }
                leaf start-label {
                  type rt-types:mpls-label;
                  must '. <= ../end-label' {
                    error-message
                      "The start-label must be less than or equal "
                    + "to end-label";
                  }
                  description
                    "Label-range start.";
                }
                leaf end-label {
                  type rt-types:mpls-label;
                  must '. >= ../start-label' {
                    error-message
                      "The end-label must be greater than or equal "
                    + "to start-label";
                  }
                  description
                    "Label-range end.";
                }
                uses sl-pol-ss-mpls-label-location;
              }
            }
          }
        }
        container ipv4 {
          description
            "Container for IPv4 Slice Selector.";
          leaf-list destination-prefix {
            type inet:ipv4-prefix;
            description
              "IPv4 Slice Selector is picked from a specified set of
               IPv4 destination prefixes.";
          }
        }
        container ipv6 {
          description
            "Container for IPv6 Slice Selector.";
          choice ss-ipv6-type {
            description
              "Choices for IPv6 Slice Selector.";
            case ipv6-destination {
              leaf-list destination-prefix {
                type inet:ipv6-prefix;
                description
                  "IPv6 Slice Selector is picked from a specified
```

```
                    set of IPv6 destination prefixes.";
                }
              }
              case ipv6-flow-label {
                container slid-flow-labels {
                  description
                    "Container for a set of Slice IDs that are
                     encoded within the flow label.";
                  list slid-flow-label {
                    key "slid";
                    description
                      "IPv6 Slice Selector is picked from a set of
                       Slice IDs that are encoded within the flow
                       label.";
                    leaf slid {
                      type inet:ipv6-flow-label;
                      description
                        "Slice ID encoded inside the IPv6 flow label.";
                    }
                    leaf bitmask {
                      type uint32;
                      description
                        "Bitmask to extract the encoded Slice ID from
                         the IPv6 flow label.";
                    }
                  }
                }
              }
            }
          }
          leaf-list acl-ref {
            type slice-policy-acl-ref;
            description
              "Slice Selection is done based on the specified list of
               ACLs.";
          }
        }
      }
    }

    /*
     * Grouping - Slice Policy Resource Reservation
     */
    grouping sl-pol-resource-reservation {
      description
        "Grouping for slice policy resource reservation.";
      container resource-reservation {
        description
```

```
          "Container for slice policy resource reservation.";
        leaf preference {
          type uint16;
          description
            "Control plane preference for the corresponding
             slice aggregate. A higher preference
             indicates a more favorable resource
             reservation than a lower preference.";
        }
        choice max-bw-type {
          description
            "Choice of maximum bandwidth specification.";
          case bw-value {
            leaf maximum-bandwidth {
              type uint64;
              description
                "The maximum bandwidth allocated to a slice aggregate
                 on the network resources - specified as absolute
                 value.";
            }
          }
          case bw-percentage {
            leaf maximum-bandwidth-percent {
              type rt-types:percentage;
              description
                "The maximum bandwidth allocated to a slice aggregate
                 on the network resources - specified as percentage
                 of link capacity.";
            }
          }
        }
        leaf-list shared-resource-groups {
          type uint32;
          description
            "List of shared resource groups that a slice aggregate
             shares its allocated resources with.";
        }
        container protection {
          description
            "Container for slice aggregate protection reservation.";
          leaf backup-sa-id {
            type uint32;
            description
              "The ID that identifies the slice aggregate used
               for backup paths that protect primary paths in a
               specific slice aggregate.";
          }
          choice backup-bw-type {
```

```
            description
              "Choice of backup bandwidth specification.";
            case backup-bw-value {
              leaf backup-bandwidth {
                type uint64;
                description
                  "The maximum bandwidth on a network resource that
                   is allocated for backup traffic - specified as
                   absolute value.";
              }
            }
            case backup-bw-percentage {
              leaf backup-bandwidth-percent {
                type rt-types:percentage;
                description
                  "The maximum bandwidth on a network resource that
                   is allocated for backup traffic - specified as
                   percentage of the link capacity.";
              }
            }
          }
        }
      }
    }

    /*
     * Grouping - Slice policy PHB (S-PHB)
     */
    grouping sl-pol-phb {
      description
        "Grouping for S-PHB.";
      leaf phb {
        type slice-policy-phb-ref;
        description
          "Reference to a specific PHB from the list of global
           PHBs.";
      }
    }

    /*
     * Grouping - Slice policy default profile override
     */
    grouping sl-pol-override-options {
      description
        "Grouping of fields that are used to override the default
         profile of the slice policy.";
      leaf slice-selector-override {
        type slice-policy-ss-ref;
```

```
        description
          "Reference to a specific Slice Selector (different from
           default).";
      }
      leaf phb-override {
        type slice-policy-phb-ref;
        description
          "Reference to a specific PHB (different from default).";
      }
    }

    /*
     * Grouping - Standard Topology Filter
     */
    grouping sl-pol-topo-filter-standard {
      description
        "Grouping for standard topology filter.";
      choice standard-topo-type {
        description
          "Choice of standard topology filter.";
        case flex-algo {
          leaf algo-id {
            type uint8;
            description
              "Algorithm ID.";
          }
          leaf mt-id {
            type uint16;
            description
              "Multi Topology ID.";
          }
        }
        case te-topo {
          uses te-types:te-topology-identifier;
        }
      }
    }

    /*
     * Grouping - Custom Topology Filters
     */
    grouping sl-pol-topo-filter-custom {
      description
        "Grouping for custom topology filters.";
      leaf-list link-affinity {
        type string;
        description
          "Match-filter is a list of link affinities.";
```

```
      }
      leaf-list link-name {
        type string;
        description
          "Match-filter is a list of link names.";
      }
      leaf-list node-prefix {
        type inet:ip-prefix;
        description
          "Match-filter is a list of node IDs.";
      }
      leaf-list as {
        type inet:as-number;
        description
          "Match-filter is a list of AS numbers.";
      }
    }

    /*
     * Grouping - Member Topologies
     */
    grouping sl-pol-member-topologies {
      description
        "Grouping for member topologies.";
      container member-topologies {
        description
          "Container for member topologies.";
        list member-topology {
          key "topology-filter";
          description
            "List of member topologies.";
          leaf topology-filter {
            type slice-policy-topo-filter-ref;
            description
              "Reference to a specific topology filter from the list
               of global topology filters.";
          }
          uses sl-pol-override-options;
        }
      }
    }

    /*
     * Grouping - Per-Hop Behaviors (PHBs)
     */
    grouping sl-pol-phbs {
      description
        "Grouping for PHBs.";
```

```
       container phbs {
         description
           "Container for PHBs.";
         list phb {
           key "id";
           description
             "List of PHBs.";
           leaf id {
             type uint16;
             description
               "A 16-bit ID that uniquely identifies the PHB.";
           }
           choice profile-type {
             description
               "Choice of PHB profile type.";
             case profile {
               description
                 "Generic PHB profile available on the network
                  element.";
               leaf profile {
                 type string;
                 description
                   "Generic PHB profile identifier.";
               }
             }
             case custom-profile {
               description
                 "Custom PHB profile.";
               choice guaranteed-rate-type {
                 description
                   "Guaranteed rate is the committed information rate
                    (CIR) of the slice aggregate. The guaranteed rate
                    also determines the amount of excess (extra)
                    bandwidth that a group of slice aggregates can
                    share. Extra bandwidth is allocated among the
                    group in proportion to the guaranteed rate of
                    each slice aggregate.";
                 case rate {
                   leaf guaranteed-rate {
                     type uint64;
                     description
                       "Guaranteed rate specified as absolute value.";
                   }
                 }
                 case percentage {
                   leaf guaranteed-rate-percent {
                     type rt-types:percentage;
                     description
```

```
                    "Guaranteed rate specified in percentage.";
                }
              }
            }
            choice shaping-rate-type {
              description
                "Shaping rate is the maximum bandwidth of the slice
                 aggregate; the peak information rate (PIR) of a
                 slice aggregate.";
              case rate {
                leaf shaping-rate {
                  type uint64;
                  description
                    "Shaping rate specified as absolute value.";
                }
              }
              case percentage {
                leaf shaping-rate-percent {
                  type rt-types:percentage;
                  description
                    "Shaping rate specified in percentage.";
                }
              }
            }
            container classes {
              description
                "Container for classes.";
              list class {
                key class-id;
                description
                  "List of classes.";
                leaf class-id {
                  type string;
                  description
                    "A string to uniquely identify a class.";
                }
                leaf direction {
                  type identityref {
                    base s-phb-class-direction;
                  }
                  description
                    "Class direction.";
                }
                leaf priority {
                  type identityref {
                    base s-phb-class-priority;
                  }
                  description
```

```
        "Priority of the class scheduler. Only one slice
         aggregate class queue can be set as a
         strict-high priority queue. Strict-high
         priority allocates the scheduled bandwidth to
         the queue before any other queue receives
         bandwidth. Other queues receive the bandwidth
         that remains after the strict-high queue has
         been serviced.";
      }
      choice guaranteed-rate-type {
        description
          "Guaranteed Rate is the Committed information
           rate (CIR) of slice aggregate class - specified
           as absolute value or percentage.";
        case rate {
          leaf guaranteed-rate {
            type uint64;
            description
              "Guaranteed rate specified as absolute
               value.";
          }
        }
        case percentage {
          leaf guaranteed-rate-percent {
            type rt-types:percentage;
            description
              "Guaranteed rate specified in percentage.";
          }
        }
      }
      leaf drop-probability {
        type identityref {
          base s-phb-class-drop-probability;
        }
        description
          "Drop probability applied to packets exceeding
           the CIR of the class queue.";
      }
      choice maximum-bandwidth-type {
        description
          "Maximum bandwidth is the Peak information
           rate (PIR) of slice aggregate class - specified
           as absolute value or percentage.";
        case rate {
          leaf maximum-bandwidth {
            type uint64;
            description
              "Maximum bandwidth specified as absolute
```

```
                        value.";
                    }
                  }
                  case percentage {
                    leaf maximum-bandwidth-percent {
                      type rt-types:percentage;
                      description
                        "Maximum bandwidth specified as percentage.";
                    }
                  }
                }
                choice delay-buffer-size-type {
                  description
                    "Size of the queue buffer as a percentage of the
                     dedicated buffer space - specified as value or
                     percentage.";
                  case value {
                    leaf delay-buffer-size {
                      type uint64;
                      description
                        "Delay buffer size.";
                    }
                  }
                  case percentage {
                    leaf delay-buffer-size-percent {
                      type rt-types:percentage;
                      description
                        "Delay buffer size specified as percentage.";
                    }
                  }
                }
              }
            }
          }
        }
      }
    }

    /*
     * Grouping - Topology Filters
     */
    grouping sl-pol-topology-filters {
      description
        "Grouping for topology filters.";
      container topology-filters {
        description
          "Container for topology filters.";
```

```
        list topology-filter {
          key "name";
          description
            "List of topology filters.";
          leaf name {
            type string;
            description
              "A string that uniquely identifies the topology filter.";
          }
          choice topology-filter-type {
            description
              "Choice of topology filter type.";
            case standard-topology {
              uses sl-pol-topo-filter-standard;
            }
            case custom-topology {
              container include-any {
                description
                  "Include-any filters.";
                uses sl-pol-topo-filter-custom;
              }
              container include-all {
                description
                  "Include-all filters.";
                uses sl-pol-topo-filter-custom;
              }
              container exclude {
                description
                  "Exclude filters.";
               uses sl-pol-topo-filter-custom;
              }
            }
          }
        }
      }
    }

    /*
     * Grouping - Slice Policies
     */
    grouping sl-policies {
      description
        "Grouping for slice policies.";
      container slice-policies {
        description
          "Container for slice policies.";
        list slice-policy {
          key "name";
```

```
          unique "sa-id";
          description
            "List of slice policies.";
          leaf name {
            type string;
            description
              "A string that uniquely identifies the slice policy.";
          }
          leaf sa-id {
            type uint32;
            description
              "A 32-bit ID that uniquely identifies the slice
               aggregate created by the enforcement of this slice
               policy.";
          }
          uses sl-pol-resource-reservation;
          uses sl-pol-slice-selector;
          uses sl-pol-phb;
          uses sl-pol-member-topologies;
        }
      }
    }

    /*
     * Top-level container - Network Slicing
     */
    container network-slicing {
      presence "Enable network slicing.";
      description
        "Top-level container for network slicing specific constructs
         on a slice policy capable network entity.";
      uses sl-pol-phbs;
      uses sl-pol-topology-filters;
      uses sl-policies;
    }
  }
  <CODE ENDS>
```

3. Acknowledgements

The authors would like to thank Krzysztof Szarkowicz for his input from discussions.

4. Contributors

The following individuals contributed to this document:

Colby Barth

Juniper Networks
Email: cbarth@juniper.net

Srihari R. Sangli
Juniper Networks
Email: ssangli@juniper.net

Chandra Ramachandran
Juniper Networks
Email: csekar@juniper.net

5. IANA Considerations

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

URI: urn:ietf:params:xml:ns:yang:ietf-network-slice-phd
Registrant Contact: The TEAS WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-network-slice-phd
namespace: urn:ietf:params:xml:ns:yang:ietf-network-slice-phd
prefix: ns-phd
reference: RFCXXXX

6. 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.

The data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default) 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:

* "/network-slicing/phbs": This subtree specifies the configurations for slice policy per-hop behaviors. By manipulating these data nodes, a malicious attacker may cause unauthorized and improper behavior to be provided for the slice aggregate traffic on the network element.

* "/network-slicing/topology-filters": This subtree specifies the configurations for slice policy topology filters. By manipulating these data nodes, a malicious attacker may cause unauthorized and improper behavior to be provided for the slice aggregate traffic on the network element.

* "/network-slicing/slice-policies": This subtree specifies the configurations for slice policies on a given network element. By manipulating these data nodes, a malicious attacker may cause unauthorized and improper behavior to be provided for the slice aggregate traffic on the network element.

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:

* "/network-slicing/phbs": Unauthorized access to this subtree can disclose the slice policy PHBs defined on the network element.

* "/network-slicing/topology-filters": Unauthorized access to this subtree can disclose the slice policy topology filters on the network element.

* "/network-slicing/slice-policies": Unauthorized access to this subtree can disclose the slice policy definitions on the network element.

7. References

7.1. Normative References

[I-D.bestbar-teas-ns-packet]
Saad, T., Beeram, V., Wen, B., Ceccarelli, D., Halpern, J., Peng, S., Chen, R., and X. Liu, "Realizing Network Slices in IP/MPLS Networks", draft-bestbar-teas-ns-packet-01 (work in progress), December 2020.

[I-D.ietf-teas-ietf-network-slice-definition]
Rokui, R., Homma, S., Makhijani, K., Contreras, L., and J. Tantsura, "Definition of IETF Network Slices", draft-ietf-teas-ietf-network-slice-definition-00 (work in progress), January 2021.

[I-D.nsdt-teas-ns-framework]
Gray, E. and J. Drake, "Framework for Transport Network Slices", draft-nsdt-teas-ns-framework-04 (work in progress), July 2020.

[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>.

[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>.

[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>.

7.2. Informative References

[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>.

Appendix A. Complete Model Tree Structure

```
module: ietf-slice-policy
  +--rw network-slicing!
     +--rw phbs
     |  +--rw phb* [id]
     |     +--rw id                                    uint16
     |     +--rw (profile-type)?
     |        +--:(profile)
     |        |  +--rw profile?                        string
     |        +--:(custom-profile)
     |           +--rw (guaranteed-rate-type)?
     |           |  +--:(rate)
     |           |  |  +--rw guaranteed-rate?          uint64
     |           |  +--:(percentage)
     |           |     +--rw guaranteed-rate-percent?
     |           |             rt-types:percentage
     |           +--rw (shaping-rate-type)?
     |           |  +--:(rate)
     |           |  |  +--rw shaping-rate?             uint64
     |           |  +--:(percentage)
     |           |     +--rw shaping-rate-percent?
     |           |             rt-types:percentage
     |           +--rw classes
     |              +--rw class* [class-id]
     |                 +--rw class-id
     |                 |       string
     |                 +--rw direction?
     |                 |       identityref
```

```
|                +--rw priority?
|                |       identityref
|                +--rw (guaranteed-rate-type)?
|                |  +--:(rate)
|                |  |  +--rw guaranteed-rate?
|                |  |          uint64
|                |  +--:(percentage)
|                |     +--rw guaranteed-rate-percent?
|                |             rt-types:percentage
|                +--rw drop-probability?
|                |       identityref
|                +--rw (maximum-bandwidth-type)?
|                |  +--:(rate)
|                |  |  +--rw maximum-bandwidth?
|                |  |          uint64
|                |  +--:(percentage)
|                |     +--rw maximum-bandwidth-percent?
|                |             rt-types:percentage
|                +--rw (delay-buffer-size-type)?
|                   +--:(value)
|                   |  +--rw delay-buffer-size?
|                   |          uint64
|                   +--:(percentage)
|                      +--rw delay-buffer-size-percent?
|                              rt-types:percentage
|  +--rw topology-filters
|  |  +--rw topology-filter* [name]
|  |     +--rw name                                  string
|  |     +--rw (topology-filter-type)?
|  |        +--:(standard-topology)
|  |        |  +--rw (standard-topo-type)?
|  |        |     +--:(flex-algo)
|  |        |     |  +--rw algo-id?                 uint8
|  |        |     |  +--rw mt-id?                   uint16
|  |        |     +--:(te-topo)
|  |        |        +--rw te-topology-identifier
|  |        |           +--rw provider-id?   te-global-id
|  |        |           +--rw client-id?     te-global-id
|  |        |           +--rw topology-id?   te-topology-id
|  |        +--:(custom-topology)
|  |           +--rw include-any
|  |           |  +--rw link-affinity*   string
|  |           |  +--rw link-name*       string
|  |           |  +--rw node-prefix*     inet:ip-prefix
|  |           |  +--rw as*              inet:as-number
|  |           +--rw include-all
|  |           |  +--rw link-affinity*   string
|  |           |  +--rw link-name*       string
```

```
        |           |  +--rw node-prefix*     inet:ip-prefix
        |           |  +--rw as*              inet:as-number
        |           +--rw exclude
        |              +--rw link-affinity*   string
        |              +--rw link-name*       string
        |              +--rw node-prefix*     inet:ip-prefix
        |              +--rw as*              inet:as-number
        +--rw slice-policies
           +--rw slice-policy* [name]
              +--rw name                    string
              +--rw sa-id?                  uint32
              +--rw resource-reservation
              |  +--rw preference?                        uint16
              |  +--rw (max-bw-type)?
              |  |  +--:(bw-value)
              |  |  |  +--rw maximum-bandwidth?          uint64
              |  |  +--:(bw-percentage)
              |  |     +--rw maximum-bandwidth-percent?
              |  |             rt-types:percentage
              |  +--rw shared-resource-groups*           uint32
              |  +--rw protection
              |     +--rw backup-sa-id?                    uint32
              |     +--rw (backup-bw-type)?
              |        +--:(backup-bw-value)
              |        |  +--rw backup-bandwidth?          uint64
              |        +--:(backup-bw-percentage)
              |           +--rw backup-bandwidth-percent?
              |                   rt-types:percentage
              +--rw slice-selectors
              |  +--rw slice-selector* [index]
              |     +--rw index      uint16
              |     +--rw mpls
              |     |  +--rw (ss-mpls-type)?
              |     |     +--:(label-value)
              |     |     |  +--rw label?
              |     |     |  |       rt-types:mpls-label
              |     |     |  +--rw label-position?         identityref
              |     |     |  +--rw label-position-offset?  uint8
              |     |     +--:(label-ranges)
              |     |        +--rw label-range* [index]
              |     |           +--rw index                    string
              |     |           +--rw start-label?
              |     |           |       rt-types:mpls-label
              |     |           +--rw end-label?
              |     |           |       rt-types:mpls-label
              |     |           +--rw label-position?
              |     |           |       identityref
              |     |           +--rw label-position-offset?   uint8
```

```
               |     +--rw ipv4
               |     |  +--rw destination-prefix*   inet:ipv4-prefix
               |     +--rw ipv6
               |     |  +--rw (ss-ipv6-type)?
               |     |     +--:(ipv6-destination)
               |     |     |  +--rw destination-prefix*
               |     |     |          inet:ipv6-prefix
               |     |     +--:(ipv6-flow-label)
               |     |        +--rw slid-flow-labels
               |     |           +--rw slid-flow-label* [slid]
               |     |              +--rw slid       inet:ipv6-flow-label
               |     |              +--rw bitmask?   uint32
               |     +--rw acl-ref*   slice-policy-acl-ref
               +--rw phb?                    slice-policy-phb-ref
               +--rw member-topologies
                  +--rw member-topology* [topology-filter]
                     +--rw topology-filter
                     |       slice-policy-topo-filter-ref
                     +--rw slice-selector-override?   slice-policy-ss-ref
                     +--rw phb-override?
                             slice-policy-phb-ref
```

Authors' Addresses

Tarek Saad
Juniper Networks

Email: tsaad@juniper.net

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

Daniele Ceccarelli
Ericsson

Email: daniele.ceccarelli@ericsson.com

Shaofu Peng
ZTE Corporation

Email: peng.shaofu@zte.com.cn

Ran Chen
ZTE Corporation

Email: chen.ran@zte.com.cn

Luis M. Contreras
Telefonica

Email: luismiguel.contrerasmurillo@telefonica.com

Xufeng Liu
Volta Networks

Email: xufeng.liu.ietf@gmail.com