Abstract

This document defines a YANG data model that can be used to configure and manage Segment Routing extensions in BGP.

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 January 8, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

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

1. Introduction

YANG [RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces (e.g. ReST) [RFC8040] and encodings other than XML (e.g. JSON) [RFC7951] are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interfaces, such as CLI and programmatic APIs.

This document defines the YANG model for Segment Routing specific extensions in BGP.
1.1. Requirements Language

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

2. BGP Segment Routing Yang model

2.1. Overview

Segment Routing (SR), as defined in [RFC8402], leverages the source routing paradigm where a node steers a packet through an ordered list of instructions, called segments. SR, thus, allows enforcing a flow through any topological path and/or service chain while maintaining per-flow state only at the ingress nodes to the SR domain.

When applied to ipv6 data-plane (i.e. SRv6), the ordered set of instructions are realized via SRv6 SIDs. The various functions and behaviors corresponding to network programming using SRv6 are specified in [I-D.ietf-spring-srv6-network-programming].

This document defines Yang model for the Segment Routing extensions applicable for BGP as following:

- Prefix sid extensions in the context of SR MPLS, as described in [I-D.ietf-idr-bgp-prefix-sid].
- Egress Peer Engineering (EPE) as described in [I-D.ietf-spring-segment-routing-central-epe].

The Yang extensions proposed in this model augment the base BGP model defined in [I-D.ietf-idr-bgp-model].

Note: Base BGP model does not have a common structure for BGP RIB. The placeholder containers defined in this model can be removed once base BGP model has the BGP RIB structure.

The modeling in this document complies with the Network Management Datastore Architecture (NMDA) [RFC8342]. The operational state data is combined with the associated configuration data in the same hierarchy [RFC8407]. When protocol states are retrieved from the NMDA operational state datastore, the returned states cover all "config true" (rw) and "config false" (ro) nodes defined in the schema.
2.2. SR Prefix SID (SR MPLS)

Prefix SID attribute in BGP in the context of SR MPLS, carries the label index and SRGB block information.

- The configuration to attach the label index is modeled as a new route-policy set action. BGP policy actions from the BGP policy module defined in base BGP yang model [I-D.ietf-idr-bgp-model] are augmented for this purpose.

- The configuration related to SR Mapping Server in the context of BGP prefix SID, is TBD.

- Prefix SID attribute received with the BGP route is modeled under BGP AF mode for select address families. This information is applicable per route.

2.3. Egress Peer Engineering

Egress Peer Engineering (EPE) in the context of Segment Routing is described in [I-D.ietf-spring-segment-routing-central-epe]. EPE is enabled in the context of BGP neighbor session. Three different types of EPE SIDs namely, Peer node SID, Peer adjacency SID and Peer set SID correspond to the segments required for source routed inter domain paths. EPE SID(s) for each type above, can be statically configured or dynamically allocated by the node. Further, FRR backup policy and backup SID can be specified per EPE. The configuration and state for the EPE parameters is modeled by augmenting the neighbor container defined in the base BGP model [I-D.ietf-idr-bgp-model]. The configuration and the applied config state are applicable for neighbor-groups as well.

3. Yang Tree

3.1. SR Prefix Sid (SR MPLS)

module: ietf-bgp-sr
  ---ro routes
  "---ro route* [prefix neighbor add-path-id]
     ---ro prefix  union
     ---ro neighbor inet:ip-address
  "---ro add-path-id uint32
  ---ro routes
  "---ro route* [prefix neighbor add-path-id]
     ---ro prefix  union
     ---ro neighbor inet:ip-address
++--ro add-path-id  uint32
++--ro routes
++--ro route* [prefix neighbor add-path-id]
++--ro prefix union
++--ro neighbor inet:ip-address
++--ro add-path-id  uint32
++--ro prefix-sid
++--ro label-index?  uint32
++--ro originator-srgb
++--ro srgb-ranges* [srgb-min srgb-max]
++--ro srgb-min  rt-types:mpls-label
++--ro srgb-max  rt-types:mpls-label
++--ro routes
++--ro route* [prefix neighbor add-path-id]
++--ro prefix union
++--ro neighbor inet:ip-address
++--ro add-path-id  uint32
++--ro prefix-sid
++--ro label-index?  uint32
++--ro originator-srgb
++--ro srgb-ranges* [srgb-min srgb-max]
++--ro srgb-min  rt-types:mpls-label
++--ro srgb-max  rt-types:mpls-label
++--ro routes
++--ro route* [rd prefix neighbor add-path-id]
++--ro rd   rt-types:route-distinguisher
++--ro prefix union
++--ro neighbor inet:ip-address
++--ro add-path-id  uint32
++--ro routes
++--ro route* [rd prefix neighbor add-path-id]
++--ro rd   rt-types:route-distinguisher
++--ro prefix union
++--ro neighbor inet:ip-address
++--ro add-path-id  uint32
++--rw set-label-index?  uint32
...
3.2. Egress Peer Engineering

Egress Peer Engineering Yang Tree applicable to neighbor and peer-group containers
module: ietf-bgp-sr

augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/bgp:bgp:neighbors/bgp:neighbor:
  ++--rw egress-peer-engineering
  |   ++--rw sid-allocation-type?  enumeration
  |   ++--rw explicit-sid?        sid-type
  |   ++--ro allocated-sid?       sid-type
  |   ++--rw peer-set-name?       string
  |   ++--rw backup
  |   |   ++--ro active?           boolean
  |   |   ++--rw backup-type?       enumeration
  |   |   ++--rw backup-peer?       inet:ip-address
  |   |   ++--ro backup-sid?        sid-type
  |   ++--rw peer-adjacency* [first-hop-ipaddress]
  |   |   ++--rw first-hop-ipaddress  inet:ip-address
  |   |   ++--rw first-hop-interface? string
  |   |   ++--rw sid-allocation-type? enumeration
  |   |   ++--rw explicit-sid?       sid-type
  |   |   ++--ro allocated-sid?      sid-type
  |   |   ++--rw backup
  |   |   |   ++--ro active?          boolean
  |   |   |   ++--rw backup-type?      enumeration
  |   |   |   ++--rw backup-peer?      inet:ip-address
  |   |   |   ++--rw backup-sid?       sid-type

augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/bgp:bgp:peer-groups/bgp:peer-group:
  ++--rw egress-peer-engineering
  |   ++--rw sid-allocation-type?  enumeration
  |   ++--rw explicit-sid?        sid-type
  |   ++--ro allocated-sid?       sid-type
  |   ++--rw peer-set-name?       string
  |   ++--rw backup
  |   |   ++--ro active?           boolean
  |   |   ++--rw backup-type?       enumeration
  |   |   ++--rw backup-peer?       inet:ip-address
  |   |   ++--rw backup-sid?        sid-type
  |   ++--rw peer-adjacency* [first-hop-ipaddress]
  |   |   ++--rw first-hop-ipaddress  inet:ip-address
  |   |   ++--rw first-hop-interface? string
  |   |   ++--rw sid-allocation-type? enumeration
  |   |   ++--rw explicit-sid?       sid-type
  |   |   ++--ro allocated-sid?      sid-type
  |   |   ++--rw backup
  |   |   |   ++--ro active?          boolean
  |   |   |   ++--rw backup-type?      enumeration
  |   |   |   ++--rw backup-peer?      inet:ip-address
  |   |   |   ++--rw backup-sid?       sid-type

...
4. Yang Module

<CODE BEGINS> file "ietf-bgp-sr@2019-07-07.yang"

module ietf-bgp-sr {

    prefix ietf-bgp-sr ;

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

    import ietf-routing {
        prefix "rt";
    }

    import ietf-inet-types {
        prefix inet;
    }

    import ietf-routing-policy {
        prefix rpol;
    }

    import ietf-bgp {
        prefix bgp;
    }

    import ietf-srv6-types {
        prefix srv6-types;
    }

    organization
       "IETF Spring Working Group";

    contact
       "Spring working group - spring@ietf.org";

    description
        "This YANG module defines a data model to configure and
         manage segment routing extensions in BGP.

        Terms and Acronyms

        AF : Address Family"
BGP (bgp) : Border Gateway Protocol
EPE : Egress Peer Engineering
SR : Segment Routing
SID : Segment Identifier
VPN : Virtual Private Network
VRF : Virtual Routing and Forwarding

revision 2018-06-26 {
  description
    "Initial revision" ;
  reference "";
}

// New identities and typedefs for SR extensions
//
// Sid type union
typedef sid-type {
  type union {
    type rt-types:mpls-label;
    type srv6-types:srv6-sid;
  }
  description "Type definition for Segment Identifier. This is
    a union type which can be either a SR MPLS SID in the
    form of a label, or a SRv6 SID in the form of
    an IPv6 address.";
  reference "TBD";
}

// SR Prefix SID related groupings
//
// Prefix SID attribute state in a route
grouping sr-route-prefix-sid {
  description "SR Prefix SID attribute associated with BGP Route";
  container prefix-sid {
    description "Prefix SID attribute";
    leaf label-index {
type uint32;
description "Label Index TLV carried with Prefix SID";
}

container originator-srgb {
description "SRGB info of the originating node,
as signaled in the originator SRGB TLV";
list srgb-ranges {
key "srgb-min srgb-max";
description "Concatenated ranges building the SRGB block";

leaf srgb-min {
    type rt-types:mpls-label;
description "Range min";
}
leaf srgb-max {
    type rt-types:mpls-label;
description "Range max";
}
}
}

//
// SR Egress Peer Engineering (EPE) related groupings
//
grouping epe-sid-alloc-mode {
    description "Common grouping for EPE mode and SID";
    leaf sid-allocation-type {
        type enumeration {
            enum EXPLICIT {
                description "EPE SID is configured";
            }
            enum DYNAMIC {
                description "EPE SID is generated by node";
            }
        }
        default "DYNAMIC";
description "SID allocation mode specifies whether the EPE SID
is explicitly configured value, or a dynamically allocated
value by the node. This applicable for EPE peer SID, EPE peer
adjacency SID and Peer set SID, depending on the context
it is configured.";
    }

leaf explicit-sid {
    //when "./mode = 'EXPLICIT'";
type sid-type;
description "Explicitly configured EPE SID value, when the sid-allocation-type is EXPLICIT";
}

leaf allocated-sid {
    type sid-type;
    config false;
    description "EPE SID value allocated by the node. When the sid allocation type is DYNAMIC, this would be a SID allocated by the node. In the case of EXPLICIT allocation type, this would typically be the explicit sid value configured by the user";
}

grouping epe-backup-info {
    description "Parameters for EPE backup SID selection";
    container backup {
        description "Backup policy for this EPE";
        leaf active {
            type boolean;
            config false;
            description "Boolean indicating if the backup as per requested policy is active for this EPE. Typically when EPE Peer, Link or Set is down, backup SID as per backup policy, would become active";
        }
        leaf backup-type {
            type enumeration {
                enum PeerNodeSid {
                    description "Backup via another Peer Node SID to the same AS. A Peer identifier is also required when this backup-type is selected";
                }
                enum PeerAdjSid {
                    description "Backup via remaining Peer Adjacencies to the same peer";
                }
                enum PeerSetSid {
                    description "Backup via Remaining PeerNode SIDs in the same PeerSet";
                }
                enum IGP {
                    description "Pop the EPE SID and perform IP lookup";
                }
            } default "IGP";
            description "Type of the backup for this EPE";
        }
    }
}
leaf backup-peer {
  //when "../backup-type = 'PeerNodeSid'"
  type inet:ip-address;
  description "Peer identifier for the case when backup
type is PeerNodeSid";
}

leaf backup-sid {
  type sid-type;
  description "Backup SID (of a EPE Peer, Peer Adjacency or Peer-Set) to be
  used as backup for this EPE";
}

grouping epe-config {
  description "Egress Peer Engineering (EPE) config grouping";
  container egress-peer-engineering {
    description "Egress Peer Engineering (EPE) config under BGP Peer";
    uses epe-sid-alloc-mode;
    leaf peer-set-name {
      type string;
      description "Make this EPE peer a member of the named
      Peer Set.";
    }
    uses epe-backup-info;
    list peer-adjacency {
      key first-hop-ipaddress;
      description "EPE parameters for the adjacency links
      over which multi-hop peering is setup";
      leaf first-hop-ipaddress {
        type inet:ip-address;
        description "First hop IP address of the link";
      }
      leaf first-hop-interface {
        type string;
        config false;
        description "The interface corresponding to the link";
      }
    }
  }
}
uses epe-sid-alloc-mode;
uses epe-backup-info;

}

}

grouping route-key-leafs {
  description "Grouping for key leafs identifying a route";
  leaf prefix {
    type union {
      type inet:ip-prefix;
      type string;
    }
    description "BGP Prefix. This is a temp definition to cover ip-prefix and other NLRI formats. Import the type once defined in base BGP RIB model";
  }
  leaf neighbor {
    type inet:ip-address;
    description "BGP Neighbor";
  }
  leaf add-path-id {
    type uint32;
    description "Add-path ID";
  }
}

grouping common-bgp-route-grouping {
  description "BGP route list";
  container routes {
    config false;
    description "BGP Route in local RIB";
    list route {
      key "prefix neighbor add-path-id";
      description "BGP route list";
      uses route-key-leafs;
    }
  }
}

grouping common-bgp-vpn-route-grouping {
  description "BGP route list";
  container routes {
    config false;
    description "BGP VPN Route in local RIB";
  }
}
list route {
  key "rd prefix neighbor add-path-id";
  description "Route List";

  leaf rd {
    type rt-types:route-distinguisher;
    description "Route Distinguisher";
  }
  uses route-key-leafs;
}

//
// BGP Specific Parameters
//
// Augment AF with route list

augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/* +
  "bgp:bgp/bg榕/global/bg榕:afi-safis/bg榕:afi-safи/bg榕:ipv4-unicast" {
  description
  "Augment BGP SAFI route";
  uses common-bgp-route-grouping;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/* +
  "bgp:bgp/bg榕/global/bg榕:afi-safis/bg榕:afi-safи/bg榕:ipv6-unicast" {
  description
  "Augment BGP SAFI route";
  uses common-bgp-route-grouping;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/* +
  "bgp:bgp/bg榕/global/bg榕:afi-safis/bg榕:afi-safи/bg榕:ipv4-labeled-unicast" {
  description
  "Augment BGP SAFI route";
  uses common-bgp-route-grouping;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/* +
  "bgp:bgp/bg榕/global/bg榕:afi-safis/bg榕:afi-safи/bg榕:ipv6-labeled-unicast" {
  description
  "Augment BGP SAFI route";
  uses common-bgp-route-grouping;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/* +
  "bgp:bgp/bg榕/global/bg榕:afi-safis/bg榕:afi-safи/bg榕:l3vpn-ipv4-unicast" {
  description
  "Augment BGP SAFI route";
  uses common-bgp-vpn-route-grouping;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp/global/bgp:afi-safis/bgp:afi-safi/bgp:13vpn-ipv6-unicast" {
    description
    "Augment BGP SAFI route";
    uses common-bgp-vpn-route-grouping;
  }

// SR Prefix SID Related.
// Prefix SID label index config via Route Policy
augment "/rpol:routing-policy/" +
  "rpol:policy-definitions/rpol:policy-definition/" +
  "rpol:statements/rpol:statement/" +
  "rpol:actions/rpol:policy-result" {
  description
  "BGP policy actions to set label index";
  leaf set-label-index {
    type uint32;
    description "Label Index";
  }
}

// Prefix SID label in SAFI route
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp/global/bgp:afi-safis/bgp:afi-safi/bgp:ipv4-labeled-unicast/" +
  "ietf-bgp-sr:routes/ietf-bgp-sr:route" {
  description
  "Augment BGP AF Table for SR prefix sid Labels info";
  uses sr-route-prefix-sid;
}

augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp/global/bgp:afi-safis/bgp:afi-safi/bgp:ipv6-labeled-unicast/" +
  "ietf-bgp-sr:routes/ietf-bgp-sr:route" {
  description
  "Augment BGP AF Table for SR prefix sid Labels info";
  uses sr-route-prefix-sid;
}

// Egress Peer Engineering (EPE) related.
// EPE config under neighbor
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp:neighbors/bgp:neighbor" {
  description
  "Egress Peer Engineering data";
  uses epe-config;
}

augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp:peer-groups/bgp:peer-group" {

5. Contributors

Dhanendra Jain
Cisco Systems
US
Email: dhanendra.ietf@gmail.com

Zhichun Jiang
Cisco Systems
US
Email: zcjiang@tencent.com

Zafar Ali
Cisco Systems
US
Email: zali@cisco.com

Sharmila Palani
Cisco Systems
US
Email: spalani@cisco.com

6. IANA Considerations

7. Security Considerations

The transport protocol used for sending the BGP Segment Routing data MUST support authentication and SHOULD support encryption. The data-model by itself does not create any security implications.

This draft does not change any underlying security issues inherent in [I-D.ietf-idr-bgp-model].
8. Acknowledgements

TBD.

9. References

9.1. Normative References

[I-D.ietf-idr-bgp-model]

[I-D.ietf-idr-bgp-prefix-sid]


9.2. Informative References
[I-D.ietf-spring-segment-routing-central-epe]

[I-D.ietf-spring-srv6-network-programming]


Authors’ Addresses
Kausik Majumdar
Cisco
170 W. Tasman Drive
San Jose, CA  95134
USA

Email: kmajumda@cisco.com

Kamran Raza
Cisco
2000 Innovation Drive
Kanata, ON  K2K-3E8
CA

Email: skraza@cisco.com