Abstract

This document defines a YANG data model that can be used to configure and manage Segment Routing TE 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 TE 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 Traffic Engineering 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 TE extensions applicable for BGP as following:

- BGP signaled SR Policy as described in [I-D.ietf-idr-segment-routing-te-policy].
- Automatic Steering as described in [I-D.ietf-spring-segment-routing-policy] and [I-D.ietf-idr-segment-routing-te-policy].

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 Policy

Architecture for SR Policies is described in [I-D.ietf-spring-segment-routing-policy]. BGP Signaled SR Policies are described in the [I-D.ietf-idr-segment-routing-te-policy]. Following Yang extensions for SR Policy configuration and state data are applicable:

- Addition of identities extending the BGP-AFI-SAFI base identity. This is to add two new address families namely IPv4 SR-policy and IPv6 SR-policy, as described in [I-D.ietf-idr-segment-routing-te-policy].

- BGP Signaled SR Policy candidate paths. These refer to the explicit candidate paths signaled via BGP as SAFI NLRIs, state of which is applicable in the context of BGP speaker process. This is modeled by adding SR Policy address family specific container under generic BGP afi-safi list entry defined in the base BGP model [I-D.ietf-idr-bgp-model].

- On Demand SR Policy candidate paths. These refer to the dynamic candidate paths as described in [I-D.ietf-spring-segment-routing-policy]. There are two parts to this in the context of BGP. A set of authorized SR Policy colors for on demand policy triggers, and the actual instantiated candidate paths per BGP next-hop. New containers and lists are added under BGP global mode to model this information.

- SR Policy state in the context of BGP speaker. This represents the state SR Policies (regardless of method of instantiation per candidate path). The SR Policy state is maintained in the context of BGP speaker process to realize the Automatic Steering of overlay routes. Automatic Steering extensions are described in the next section.

Note: The common parameters and datatypes for the SR Policy, currently defined in this model, should be imported from SR Policy Manager model, once available.

2.3. Automatic Steering

Automatic Steering (AS) refers to the ability to forward traffic over a SR Policy on the head-end, as described in [I-D.ietf-spring-segment-routing-policy]. When a BGP route is received with the color extended community and if the color value
matches the color of an authorized SR Policy installed on the head-end, the route is programmed to resolve over SR Policy in forwarding. Automatic Steering information associated with the BGP routes is modeled as state information per route.

TBD: The configuration parameters for Automatic Steering are yet to be added as an augmentation to the BGP route policy model. Such as, extensions for opaque color extended community in BGP policy model, and the Color Only (CO) flags controlling the Automatic Steering behavior as described in [I-D.ietf-idr-segment-routing-te-policy].

3. Yang Tree

3.1. SR Policy

On Demand Nexthop (ODN) policies triggered by BGP

augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/bgp:bgp:bgp:global:
  +--rw segment-routing
    +--rw on-demand-policies
     |  +--ro authorized-colors
     |     |  +--ro colors* [color]
     |     |      |  +--ro color uint32
     |     +--ro installed-policies
     |        +--ro sr-policy* [color end-point]
     |            |  +--ro color uint32
     |            |  +--ro end-point inet:ip-address
     +--ro policy-state
      +--ro sr-policy* [color end-point]
       |  +--ro color uint32
       |  +--ro end-point inet:ip-address
       +--ro policy-state? enumeration
       +--ro binding-sid? sid-type
       +--ro steering-disabled? empty
       +--ro ref-count? uint32

BGP Signaled Explicit SR Policies under ipv4 and ipv6 SR-Policy SAFI
3.2. Automatic Steering

Yang Tree for Automatic Steering with example of ipv4-unicast SAFI

module: ietf-bgp-srte


--- ro explicit-policies
    --- ro explicit-binding-sid
        |--- ro binding-sid? sid-type
        |--- ro strict? boolean
        |--- ro drop-on-invalid? boolean
    --- ro usable? boolean
    --- ro registered? boolean

--- ro ipv4-srpolicy
    --- ro distinguisher uint32
    --- ro color uint32
    --- ro end-point inet:ip-address
    --- ro preference? uint32
    --- ro usable? boolean
    --- ro registered? boolean

--- ro ipv6-srpolicy
    --- ro distinguisher uint32
    --- ro color uint32
    --- ro end-point inet:ip-address
    --- ro preference? uint32
    --- ro usable? boolean
    --- ro registered? boolean

--- ro routes
    --- ro route* [prefix neighbor add-path-id]
        --- ro prefix union
        --- ro neighbor inet:ip-address
        --- ro add-path-id uint32
        --- ro automatic-steering
            --- ro co-flag? enumeration
            --- ro binding-sid? sid-type
        --- ro usable? boolean
        --- ro registered? boolean
        --- ro end-point inet:ip-address

--- ro routes
    --- ro route* [prefix neighbor add-path-id]
        --- ro prefix union
        --- ro neighbor inet:ip-address
        --- ro add-path-id uint32
        --- ro automatic-steering
            --- ro co-flag? enumeration
            --- ro binding-sid? sid-type
        --- ro usable? boolean
        --- ro registered? boolean
        --- ro end-point inet:ip-address

--- ro routes
6. Yang Module

```yang
module ietf-bgp-srte {
    namespace "urn:ietf:params:xml:ns:yang:ietf-bgp-srte";
    prefix "ietf-bgp-srte";
    import ietf-routing-types {
        prefix rt-types;
    }
    import ietf-routing {
        prefix rt;
    }
    import ietf-inet-types {
        prefix inet;
    }
    import ietf-bgp {
        prefix bgp;
    }
    import ietf-bgp-types {
        prefix bgp-types;
    }
    import ietf-srv6-types {
        prefix srv6-types;
    }
    organization "IETF Inter-Domain Routing Working Group";
}
```
contact
"Inter-Domain working group - idr@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

SR : Segment Routing

SID : Segment Identifier

SRv6 : Segment Routing with IPv6 Data plane

VPN : Virtual Private Network

VRF : Virtual Routing and Forwarding

"

revision 2018-06-26 {

description
"Initial revision" ;

reference "";

}

// New identities and typedefs for SR extensions
//

// SR Policy SAFI identities
identity IPV4_SRPOLICY {
    base bgp-types:afi-safi-type;
    description
    "IPv4 SR Policy (AFI,SAFI = 1,73)";
    reference "TBD";
}

identity IPV6_SRPOLICY {
    base bgp-types:afi-safi-type;
    description
    "IPv6 SR Policy (AFI,SAFI = 2,73)";
    reference "TBD";
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 Policy Related Groupings
//
//Color and Endpoint of the SR Policy
grouping sr-policy-color-endpoint {
    description "Common grouping for SR Policy Color and Endpoint";

    leaf color {
        type uint32;
        description "Color of the policy";
    }

    leaf end-point {
        type inet:ip-address;
        description "Endpoint of the policy";
    }
}

// Authorized colors for On Demand SR Policy programming
grouping sr-odn-auth-colors {
    description "Authorized colors for On Demand (dynamic) SR Policies towards BGP nexthops";

    container authorized-colors {
        config false;
        description "Authorized colors for OnDemand (dynamic) SR policies towards BGP nexthops";

        list colors {
            key "color";
            description "List of SR Policy Colors";

            leaf color {
                type uint32;
                description "Color value";
            }
        }
    }
}
grouping sr-policy-cmn-state {
  description "Common state parameters applicable to SR Policies";
  leaf policy-state {
    type enumeration {
      enum UP {
        description "SR Policy state UP";
      }
      enum DOWN {
        description "SR Policy state DOWN";
      }
    }
    description "SR Policy forwarding state";
  }
  leaf binding-sid {
    type sid-type;
    description "Binding SID of the SR Policy";
  }
  leaf steering-disabled {
    type empty;
    description "This attribute is set if steering is disabled on this SR policy";
  }
  leaf ref-count {
    type uint32;
    description "Count of routes steering over this policy";
  }
}

// SR Policy State grouping
//

grouping sr-policy-state {
  description "SR Policy State";
  container policy-state {
    config false;
    description "SR Policy State";
    list sr-policy {
      key "color end-point";
      description "List of SR Policies";
    }
  }
}
uses sr-policy-color-endpoint;

// State of the SR Policy in BGP
uses sr-policy-cmn-state;

}
}

}

)

grouping sr-exp-policy-cp-state {
    description "State of BGP signaled SR Policy (explicit) candidate paths";
    container explicit-policies {
        config false;
        description "BGP signaled explicit SR Policies";
        list sr-policy {
            key "distinguisher color end-point";
            description "List of BGP signaled explicit SR Policies";
            leaf distinguisher {
                type uint32;
                description "Distinguisher of the SR Policy candidate path";
            }
        }
    }
    uses sr-policy-color-endpoint;

    leaf preference {
        type uint32;
        description "Preference of the SR Policy candidate path";
    }
}


container explicit-binding-sid {
    description "Explicitly supplied Binding SID for this policy";
    leaf binding-sid {
        type sid-type;
        description "Binding SID value";
    }
    leaf strict {
        type boolean;
        description "Boolean indicating that the node must use only the supplied Binding SID for this SR Policy. reference: TBD";
    }
    leaf drop-on-invalid {
        type boolean;
        description "Boolean to indicate drop upon invalid policy, behavior. This overwrites the
default behavior of fallback to IGP path, when SR Policy is (or becomes) invalid.
reference: TBD";
}
}

leaf usable {
    type boolean;
    description "Boolean to indicate that the SR Policy is
    usable on this node.
    reference: TBD"
}

leaf registered {
    type boolean;
    description "Boolean to indicate that the SR policy
    is registered with policy manager to
    install the corresponding forwarding entry";
}

// TODO: Segment Lists and other parameters from SR Policy model
// to be imported here.

}
}

grouping sr-odn-policies {
    description "SR On Demand (dynamic) SR Policies";
    container installed-policies {
        config false;
        description "BGP triggered On Demand (dynamic) SR Policies
        corresponding to the BGP nexthops";
        list sr-policy {
            key "color end-point";
            description "SR Policy list";
            uses sr-policy-color-endpoint;
        }
    }
}

// TODO: Segment Lists and other parameters from SR Policy model
// to be imported here.

}
}

// TODO: Segment Lists and other parameters from SR Policy model
// to be imported here.

}

// TODO: Segment Lists and other parameters from SR Policy model
// to be imported here.

}

"bgp:bgp/global/ietf-bgp-srte:segment-routing/" +
"ietf-bgp-srte:policy-state/ietf-bgp-srte:segment-routing/" +
"ietf-bgp-srte:policy-state/ietf-bgp-srte:sr-policy/" +
"ietf-bgp-srte:end-point/"
}
description "Color of the SR Policy being used for
Automatic Steering";
}
leaf end-point {
    type leafref {
        path "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
"bgp:bgp/global/ietf-bgp-srte:segment-routing/" +
"ietf-bgp-srte:policy-state/ietf-bgp-srte:segment-routing/" +
"ietf-bgp-srte:policy-state/ietf-bgp-srte:sr-policy/" +
"ietf-bgp-srte:end-point/"
    }
    description "End-point of the SR Policy being used
for Automatic Steering";
}
leaf co-flag {
    type enumeration {
        enum 00 { 
            description "Color-Only flag 00";
        }
        enum 01 { 
            description "Color-Only flag 01";
        }
        enum 10 { 
            description "Color-Only flag 10";
        }
    }
    default "00";
    description "Color-Only (CO) flags applicable for
Automatic Steering of this route";
}
leaf binding-sid {
    type leafref {
        path "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
"bgp:bgp/global/ietf-bgp-srte:segment-routing/" +
"ietf-bgp-srte:policy-state/ietf-bgp-srte:segment-routing/" +
"ietf-bgp-srte:policy-state/ietf-bgp-srte:sr-policy/" +
"ietf-bgp-srte:binding-sid/"
    }
    description "Binding SID of the SR Policy";
}
}
}
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/global/bgp:afi-safis/bgp:afi-safi/bgp: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/global/bgp:afi-safis/bgp:afi-safi/bgp: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/global/bgp:afi-safis/bgp:afi-safi/bgp: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/global/bgp:afi-safis/bgp:afi-safi/bgp: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/global/bgp:afi-safis/bgp:afi-safi/bgp: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:l3vpn-ipv6-unicast" {  
  description  
    "Augment BGP SAFI route";  
  uses common-bgp-vpn-route-grouping;  
}  

// SR Policy Related  
// On Demand authorized colors table  
// SR Policy state data  
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
"bgp:bgp/bgp:global" {
  description
  "Segment Routing parameters in BGP global model";
  container segment-routing {
    description "Segment Routing parameters";
    container on-demand-policies {
      description
        "Segment Routing On Demand Nexthop
        (ODN) SR Policies";
      uses sr-odn-auth-colors;
      uses sr-odn-policies;
    }
    uses sr-policy-state;
  }
}

// Steering state in overlay BGP routes
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp:bgp:global/bgp:afi-safis/" +
  "bgp:afi-safi/bgp:ipv4-unicast/ietf-bgp-srte:routes/ietf-bgp-srte:route" { 
  description
    "Augment BGP SAFI route with steering info";
  uses sr-policy-steering-state;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp:bgp:global/bgp:afi-safis/" +
  description
    "Augment BGP SAFI route with steering info";
  uses sr-policy-steering-state;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp:bgp:global/bgp:afi-safis/" +
  "bgp:afi-safi/bgp:ipv4-labeled-unicast/ietf-bgp-srte:routes/ietf-bgp-srte:route" { 
  description
    "Augment BGP SAFI route with steering info";
  uses sr-policy-steering-state;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp:bgp:global/bgp:afi-safis/" +
  "bgp:afi-safi/bgp:ipv6-labeled-unicast/ietf-bgp-srte:routes/ietf-bgp-srte:route" { 
  description
    "Augment BGP SAFI route with steering info";
  uses sr-policy-steering-state;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
  "bgp:bgp:bgp:global/bgp:afi-safis/" +
  description
    "Augment BGP SAFI route with steering info";
  uses sr-policy-steering-state;
}
description
"Augment BGP SAFI route with steering info";
uses sr-policy-steering-state;
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/
"bgp:bgp/bgp:global/bgp:afi-safis/" +
description
"Augment BGP SAFI route with steering info";
uses sr-policy-steering-state;
}

// BGP Signaled SR Policy explicit candidate paths state
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
"bgp:bgp/bgp:global/bgp:afi-safis/bgp:afi-safi" {
description "Augment IPv4 SR Policy SAFI list entry";
container ipv4-srpolicy {
when "./afi-safi-name = ‘bgp-types:IPV4_SRPOLICY’" {
description
"Include this container for IPv4 SR Policy specific configuration";
}
description "IPv4 SR Policy specific parameters";
uses sr-exp-policy-cp-state;
}
}
augment "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/" +
"bgp:bgp/bgp:global/bgp:afi-safis/bgp:afi-safi" {
description "Augment IPv6 SR Policy SAFI list entry";
container ipv6-srpolicy {
when "./afi-safi-name = ‘bgp-types:IPV6_SRPOLICY’" {
description
"Include this container for IPv6 SR Policy specific configuration";
}
description "IPv6 SR Policy specific parameters";
uses sr-exp-policy-cp-state;
}
}
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
9.2. Informative References

[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

Bruno Decraene
Orange
France

Email: bruno.decreaene@orange.com
Zhichun Jiang
Tencent

Email: zcjiang@tencent.com