Interdomain Routing
Internet-Draft
Intended status: Standards Track
Expires: 6 January 2024

M. Jethanandani
Kloud Services
K. Patel
Arrcus
S. Hares
Huawei
J. Haas
Juniper Networks

5 July 2023

YANG Model for Border Gateway Protocol (BGP-4)
draft-ietf-idr-bgp-model-17

Abstract

This document defines a YANG data model for configuring and managing BGP, including protocol, policy, and operational aspects, such as RIB, based on data center, carrier, and content provider operational requirements.

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 6 January 2024.

Copyright Notice

Table of Contents

1. Introduction

This document describes a YANG 1.1 [RFC7950] data model for the BGP-4 [RFC4271] protocol, including various protocol extensions, policy configuration, as well as defining key operational state data, including a Routing Information Base (RIB). The model is intended to be vendor-neutral, in order to allow operators to manage BGP configuration in heterogeneous environments with routers supplied by multiple vendors. The model is also intended to be readily mapped to existing implementations to facilitate support from as large a set of routing hardware and software vendors as possible. This module does not support previous versions of BGP, and cannot support establishing and maintaining state information of neighbors with previous versions of BGP.

1.1. Goals and approach

The model covers the base BGP features that are deployed across major implementations and the common BGP configurations in use across a number of operator network deployments. In particular, this model attempts to cover BGP features defined in BGP [RFC4271], BGP Communities Attribute [RFC1997], BGP Extended Communities Attributes [RFC4360], IPv6 Extended Communities Attributes [RFC5701], BGP Large Communities Attributes [RFC8092], BGP Route Reflection [RFC4456], Multiprotocol Extensions for BGP-4 [RFC4760], Autonomous System Confederations for BGP [RFC5065], BGP Route Flap Damping [RFC2439], Graceful Restart Mechanism for BGP [RFC4724], BGP Prefix Origin

Validation [RFC6811], and Advertisement of Multiple Paths in BGP [RFC7911]. It attempts to make the model extensible by using identities for well known standard community types, and the use of "raw" string to support new and experimental type definitions.

Along with configuration of base BGP features, this model also addresses policy configuration, by providing "hooks" for applying policies, and also defining BGP-specific policy features. The BGP policy features are intended to be used with the general routing policy model defined in A YANG Data Model for Routing Policy Management [RFC9067].

The model conforms to the NMDA [RFC8342] architecture. It has support for securing BGP sessions using TCP-AO [RFC5925] or TCP-MD5, and for configuring Bidirectional Forward Detection (BFD) [RFC5880] for fast next hop liveliness checking.

For the base BGP features, the focus of the model described in this document is on providing configuration and operational state information relating to:

* The global BGP instance, and neighbors whose configuration is specified individually, or templated with the use of peer-groups.

* The address families that are supported by peers, and the global configuration which relates to them.

* The policy configuration "hooks" and BGP-specific policy features that relate to a neighbor - controlling the import and export of NLRIs.

* BGP RIB contents.

As mentioned earlier, any configuration items that are deemed to be widely available in existing major BGP implementations are included in the model. Additional, more esoteric, configuration items that are not commonly used, or only available from a single implementation, are omitted from the model with an expectation that they will be available in companion modules that augment or extend the current model. This allows clarity in identifying data that is part of the vendor-neutral base model.

Where possible, naming in the model follows conventions used in available standards documents, and otherwise tries to be self-explanatory with sufficient descriptions of the intended behavior. Similarly, configuration data value constraints and default values, where used, are based on recommendations in current standards documentation, or those commonly used in multiple implementations.

Since implementations can vary widely in this respect, this version of the model specifies only a limited set of defaults and ranges with the expectation of being more prescriptive in future versions based on actual operator use.

1.2. Note to RFC Editor

This document uses several placeholder values throughout the document. Please replace them as follows and remove this note before publication.

RFC XXXX, where XXXX is the number assigned to this document at the time of publication.

2023-07-05 with the actual date of the publication of this document.

1.3. Terminology

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.

1.4. Abbreviations

Abbreviation	
AFI	Address Family Identifier
BFD	Bidirectional Forward Detection
NLRI	Network Layer Reachability Information
NMDA	Network Management Datastore Architecture
RIB	Routing Information Base
SAFI	Subsequent Address Family Identifier
VRF	Virtual Routing and Forwarding

Table 1

2. Model overview

The BGP model is defined across several YANG modules and submodules, but at a high level is organized into six elements:

* base protocol configuration -- configuration affecting BGP protocol-related operations, defined at various levels of hierarchy.

* multi-protocol configuration -- configuration affecting individual address-families within BGP Multiprotocol Extensions for BGP-4 [RFC4760].

* neighbor configuration -- configuration affecting an individual neighbor within BGP.

* neighbor multi-protocol configuration -- configuration affecting individual address-families for a neighbor within BGP.

* policy configuration -- hooks for application of the policies defined in A YANG Data Model for Routing Policy Management [RFC9067] that act on routes sent (received) to (from) peers or other routing protocols and BGP-specific policy features.

* operational state -- variables used for monitoring and management of BGP operations.

These modules also make use of standard Internet types, such as IP addresses and prefixes, autonomous system numbers, etc., defined in Common YANG Data Types [RFC6991].

2.1. BGP protocol configuration

The BGP protocol configuration model is organized hierarchically, much like the majority of router implementations. That is, configuration items can be specified at multiple levels, as shown below.

module: ietf-bgp

```
  augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol:
    +--rw bgp
       +--rw global!
       |  +--rw as                          inet:as-number
       |  +--rw identifier?                 yang:dotted-quad
       |  +--rw distance
       |  |     ...
```

```
|  +--rw confederation
|  |     ...
|  +--rw graceful-restart {bt:graceful-restart}?
|  |     ...
|  +--rw use-multiple-paths
|  |     ...
|  +--rw route-selection-options
|  |     ...
|  +--rw afi-safis
|  |     ...
|  +--rw apply-policy
|  |     ...
|  +--ro statistics
|        ...
+--rw neighbors
|  +--rw neighbor* [remote-address]
|  |     ...
|  +---n established
|  |     ...
|  +---n backward-transition
|  |     ...
|  +---x clear {bt:clear-neighbors}?
|        ...
+--rw peer-groups
|  +--rw peer-group* [name]
|        ...
+--ro rib
   +--ro attr-sets
   |     ...
   +--ro communities
   |     ...
   +--ro ext-communities
   |     ...
   +--ro ipv6-ext-communities
   |     ...
   +--ro large-communities
   |     ...
   +--ro afi-safis
         ...
```

Users may specify configuration at a higher level and have it apply to all lower-level items, or provide overriding configuration at a lower level of the hierarchy. Overriding configuration items are optional, with neighbor-specific configuration being the most specific or lowest level, followed by peer-group, and finally global. Global configuration options reflect a subset of the peer-group or neighbor-specific configuration options which are relevant to the entire BGP instance.

The model makes the simplifying assumption that most of the configuration items are available at all levels of the hierarchy. That is, very little configuration is specific to a particular level in the hierarchy, other than obvious items such as "group-name" only being available for the peer group-level config. A notable exception is for sub-address family configuration where some items are only applicable for a given AFI-SAFI combination.

In order to allow common configuration to be applied to a set of neighbors, all neighbor configuration options are available within a peer-group. A neighbor is associated with a particular peer-group through the use of a peer-group leaf (which provides a reference to a configured item in the peer-group list).

Address-family configuration is made available in multiple points within the model - primarily within the global container, where instance-wide configuration can be set (for example, global protocol parameters, the BGP best-path route selection options, or global policies relating to the address-family); and on a per-neighbor or per-peer-group basis, where address-families can be enabled or disabled, and policy associated with the parent entity applied. Within the afi-safi container, generic configuration that applies to all address-families (e.g., whether the AFI-SAFI is enabled) is presented at the top-level, with address-family specific containers made available for options relating to only that AFI-SAFI. Within the current revision of the model a generic set of address-families, and common configuration and state options are included - further work is expected to add additional parameters to this area of the model.

The model supports ipv4-unicast and ipv6-unicast address-families and defers the remaining AFI/SAFI to other or future drafts:

```
          +--rw bgp
            +--rw global!
               +--rw afi-safis
                  +--rw afi-safi* [afi-safi-name]
                     +--rw afi-safi-name              identityref
                     |
                     +--rw ipv4-unicast
                     |     ...
                     +--rw ipv6-unicast
                     |     ...
                     +--rw ipv4-labeled-unicast
                     |     ...
                     +--rw ipv6-labeled-unicast
                     |     ...
                     +--rw l3vpn-ipv4-unicast
                     |     ...
                     +--rw l3vpn-ipv6-unicast
                     |     ...
                     +--rw l3vpn-ipv4-multicast
                     |     ...
                     +--rw l3vpn-ipv6-multicast
                     |     ...
                     +--rw l2vpn-vpls
                     |     ...
                     +--rw l2vpn-evpn
                     |     ...
```

2.2. Policy configuration overview

The BGP policy configuration model augments the generic YANG routing policy model described in A YANG Data Model for Routing Policy Management [RFC9067], which represents a condition-action policy framework for routing. This model adds BGP-specific conditions (e.g., matching on the community attribute), and actions (e.g., setting local preference) to the generic policy framework.

Policies that are defined in the routing-policy model are referenced in multiple places within the model:

* within the global instance, where a policy applies to all address-families for all peers.

* on a global AFI-SAFI basis, where policies apply to all peers for a particular address-family.

* on a per-peer-group or per-neighbor basis - where the policy applies to all address-families for the particular group or neighbor.

* on a per-afi-safi basis within a neighbor or peer-group context, where the policy is specific to the AFI-SAFI for a a specific neighbor or group.

```
module: ietf-bgp-policy

  augment /rt-pol:routing-policy/rt-pol:defined-sets:
    +--rw bgp-defined-sets
          ...
  augment /rt-pol:routing-policy/rt-pol:policy-definitions
            /rt-pol:policy-definition/rt-pol:statements
            /rt-pol:statement/rt-pol:conditions:
    +--rw bgp-conditions
          ...
  augment /rt-pol:routing-policy/rt-pol:policy-definitions
            /rt-pol:policy-definition/rt-pol:statements
            /rt-pol:statement/rt-pol:actions:
    +--rw bgp-actions
          ...
```

2.3. BGP RIB overview

The RIB data model represents the BGP RIB contents. The model supports five logical RIBs per address family.

An abridged version of the tree shows the RIB portion of the tree diagram.

```
   module: ietf-bgp

     augment /rt:routing/rt:control-plane-protocols
               /rt:control-plane-protocol:
       +--rw bgp
          +--ro rib
             +--ro afi-safis
                +--ro afi-safi* [name]
                   +--ro name            identityref
                   +--ro ipv4-unicast
                   |  +--ro loc-rib
                   |  |  +--ro routes
                   |  |     +--ro route* [prefix origin path-id]
                   |  |           ...
                   |  +--ro neighbors
                   |     +--ro neighbor* [neighbor-address]
                   |        +--ro neighbor-address    inet:ip-address
                   |        +--ro adj-rib-in-pre
                   |        |     ...
                   |        +--ro adj-rib-in-post
                   |        |     ...
                   |        +--ro adj-rib-out-pre
                   |        |     ...
                   |        +--ro adj-rib-out-post
                   |              ...
                   +--ro ipv6-unicast
                      +--ro loc-rib
                      |  +--ro routes
                      |     +--ro route* [prefix origin path-id]
                      |           ...
                      +--ro neighbors
                         +--ro neighbor* [neighbor-address]
                            +--ro neighbor-address    inet:ip-address
                            +--ro adj-rib-in-pre
                            |     ...
                            +--ro adj-rib-in-post
                            |     ...
                            +--ro adj-rib-out-pre
                            |     ...
                            +--ro adj-rib-out-post
                                  ...
```

2.3.1. Local Routing

The loc-rib is the main BGP routing table for the local routing instance, containing best-path selections for each prefix. The key for the loc-rib will correspond either to a route in the adj-rib-in-post table, described below, or a route redistributed from another protocol.

2.3.2. Pre updates per-neighbor

The adj-rib-in-pre table is a per-neighbor table containing the NLRI updates received from the neighbor before any local input policy rules or filters have been applied. This can be considered the 'raw' updates from a given neighbor.

2.3.3. Post updates per-neighbor

The adj-rib-in-post table is a per-neighbor table containing the routes received from the neighbor that are eligible for best-path selection after local input policy rules have been applied.

The "best-path" leaf is attached to the route selected by the BGP Decision Process Section 9.1 of [RFC4271]. Such routes may be present in the loc-rib table, described above, when local configuration determines that the BGP best-path will be used for that destination.

2.3.4. Pre route advertisements per-neighbor

The adj-rib-out-pre table is a per-neighbor table containing routes eligible for sending (advertising) to the neighbor before output policy rules have been applied.

2.3.5. Post route advertisements per-neighbor

The adj-rib-out-post table is a per-neighbor table containing routes eligible for sending (advertising) to the neighbor after output policy rules have been applied.

3. Relation to other YANG data models

The BGP model augments the Routing Management model A YANG Data Model for Routing Management [RFC8349] which defines the notion of routing, routing protocols, and RIBs. The notion of Virtual Routing and Forwarding (VRF) is derived by using the YANG Schema Mount [RFC8528] to mount the Routing Management module under the YANG Data Model for Network Instances [RFC8529].

Similarly, it augments the A YANG Data Model for Routing Policy Management [RFC9067] to add in BGP-specific conditions and actions to the generic policy model.

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

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. Some of the subtrees and data nodes and their sensitivity/vulnerability are described here.

* The attribute 'as'. If a user is allowed to change this attribute, it will have the net effect of bringing down the entire routing instance, causing it to delete all the current routing entries, and learning new ones.

* The attribute 'identifier'. If a user is allowed to change this attribute, it will have the net effect of this routing instance re-advertising all its routes.

* The attribute 'distance'. If a user is allowed to change this attribute, it will cause the preference for routes, e.g. external vs internal to change.

* The attribute 'enabled' in the 'confederation' container. This attribute defines whether a local-AS is part of a BGP confederation.

* Finally, there are a whole set of route selection options such as 'always-compare-med', 'ignore-as-path-length' that affect the way the system picks up a particular route. Being able to change will adversely affect how the route selection happens.

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

* The list of neighbors, and their attributes. Allowing a user to read these attributes, in particular the address/port information may allow a malicious user to launch an attack at the particular address/port.

* The 'rib' container. This container contains sensitive information such as attribute sets, communities, extended communities, IPv6 extended communities, and large communities. Being able to read the contents of this container will allow a malicious user to understand how the system decide how to route a packet, and thus try to affect a change.

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

* The model allows for routes to be cleared using the 'clear' RPC operations, causing the entire RIB table to be cleared.

* The model allows for statistics to be cleared by the 'clear' RPC operation, causing all the individual statistics to be cleared.

* The model also allows for neighbors that have been learnt by the system to be cleared by using the 'clear' RPC operation.

BGP OPSEC [RFC7454] describes several policies that can be used to secure BGP. In particular, it recommends securing the underlying TCP session and to use the Generalized TTL Security Mechanism (GTSM) [RFC5082] capability to make it harder to spoof a BGP session. This module allows implementations that want to support the capability to configure a TTL value, under a feature flag. It also defines a container 'secure-session' that contains a choice of using TCP-Authentication Option (TCP-AO) [RFC5925], or Protection of BGP Sessions via the TCP MD5 Signature Option [RFC2385]. The support for MD5 has been made available for legacy reasons. However, because of vulnerabilities in MD5 as described in MD5 and HMAC-MD5 Security Considerations [RFC6151], operators are strongly encouraged to use TCP-AO or other methods to secure their BGP session.

5. IANA Considerations

This document registers six URIs and six YANG modules.

5.1. URI Registration

Following the format in the IETF XML registry [RFC3688] [RFC3688], the following registration is requested to be made:

```
URI:urn:ietf:params:xml:ns:yang:ietf-bgp
URI:urn:ietf:params:xml:ns:yang:ietf-bgp-policy
URI:urn:ietf:params:xml:ns:yang:iana-bgp-community-types
URI:urn:ietf:params:xml:ns:yang:iana-bgp-notification
URI:urn:ietf:params:xml:ns:yang:iana-bgp-rib-types
URI:urn:ietf:params:xml:ns:yang:iana-bgp-types
```

Registrant Contact: The IESG. XML: N/A, the requested URI is an XML namespace.

5.2. YANG Module Name Registration

This document registers six YANG modules in the YANG Module Names registry YANG [RFC6020].

```
            name: ietf-bgp
            namespace: urn:ietf:params:xml:ns:yang:ietf-bgp
            prefix: bgp
            reference: RFC XXXX

            name: ietf-bgp-policy
            namespace: urn:ietf:params:xml:ns:yang:ietf-bgp-policy
            prefix: bp
            reference: RFC XXXX

            name: iana-bgp-community-types
            namespace: urn:ietf:params:xml:ns:yang:iana-bgp-community-types
            prefix: bct
            reference: RFC XXXX

            name: iana-bgp-notification
            namespace: urn:ietf:params:xml:ns:yang:iana-bgp-notification
            prefix: bn
            reference: RFC XXXX

            name: iana-bgp-types
            namespace: urn:ietf:params:xml:ns:yang:iana-bgp-types
            prefix: bt
            reference: RFC XXXX

            name: iana-bgp-rib-types
            namespace: urn:ietf:params:xml:ns:yang:iana-bgp-rib-types
            prefix: brt
            reference: RFC XXXX
```

6. YANG modules

The modules comprising the BGP configuration and operational model are described by the YANG modules and submodules in the sections below.

The main module, ietf-bgp.yang, includes the following submodules:

* ietf-bgp-capabilities - defines the groupings that are common to the BGP Capabilities feature.

* ietf-bgp-common - defines the groupings that are common across more than one context (where contexts are neighbor, group, global).

* ietf-bgp-common-multiprotocol - defines the groupings that are common across more than one context, and relate to multi-protocol BGP.

* ietf-bgp-common-structure - defines groupings that are shared by multiple contexts, but are used only to create structural elements; i.e., containers (leaf nodes are defined in separate groupings).

* ietf-bgp-neighbor - groupings with data specific to the neighbor context.

* ietf-bgp-rib - grouping for representing BGP RIBs.

* ietf-bgp-rib-attributes - common data definitions for BGP attributes used in BGP RIB tables.

* ietf-bgp-rib-tables - structural data definitions for BGP routing tables.

Additionally, the above modules include the following modules:

* iana-bgp-community-types - common type and identity definitions for BGP Communities.

* ietf-bgp-policy - BGP-specific policy data definitions for use with [RFC9067] (described in more detail Section 2.2).

* iana-bgp-types - common type and identity definitions for BGP, including BGP policy.

* iana-bgp-rib-types - common type and identity definitions for BGP RIB.

7. Structure of the YANG modules

The YANG model can be subdivided between the main module for base items, types, and policy module. It references:

* BGP Communities Attribute [RFC1997]

* Route Refresh Capability for BGP-4 [RFC2918]

* NOPEER Community for BGP [RFC3765]

* BGP Extended Communities Attributes [RFC4360]

* BGP/MPLS IP Virtual Private Networks (VPNs) [RFC4364]

* BGP MED Considerations [RFC4451]
* BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN [RFC4659]
* Graceful Restart Mechanism for BGP [RFC4724]
* Multiprotocol Extentions for BGP-4 [RFC4760]
* Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling [RFC4761]
* Autonomous System Configuration for BGP [RFC5065]
* The Generalized TTL Security Mechanism (GTSM) [RFC5082]
* IPv6 Address Specific BGP Extended Community Attribute [RFC5701]
* Bidirectional Forward Detection (BFD) [RFC5880]
* Bidirectional Forward Detection for IPv4 and IPv6 (Single Hop) [RFC5881]
* Bidirectional Forwarding Detection (BFD) for Multihop Paths [RFC5883]
* The TCP Authentication Option [RFC5925]
* BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs [RFC6514]
* BGP Support for Four-Octet Autonomous System (AS) Number Space [RFC6793]
* Advertisement of Multiple Paths in BGP [RFC7911]
* BGP Large Communities Attributes [RFC8092]
* YANG Key Chain [RFC8177]
* Carrying Label Information in BGP-4 [RFC8277]
* A YANG Data Model for Routing Policy [RFC9067]
* YANG Data Model for Bidirectional Forward Detection [RFC9314]
* Transmission Control Protocol [RFC9293]

* YANG Model for Transmission Control Protocol (TCP) Configuration [I-D.ietf-tcpm-yang-tcp]

7.1. BGP module and submodules

7.1.1. BGP module

```
<CODE BEGINS> file "ietf-bgp@2023-07-05.yang"
module ietf-bgp {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-bgp";
  prefix bgp;

  /*
   * Import and Include
   */

  import ietf-routing {
    prefix rt;
    reference
      "RFC 8349: A YANG Data Model for Routing Management
       (NMDA Version).";
  }
  import ietf-routing-policy {
    prefix rt-pol;
    reference
      "RFC 9067: A YANG Data Model for Routing Policy.";
  }
  import iana-bgp-notification {
    prefix bn;
    reference
      "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
  }
  import iana-bgp-types {
    prefix bt;
    reference
      "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
  }
  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types.";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types.";
  }
```

```
    include ietf-bgp-capabilities {
      revision-date 2023-07-05;
    }
    include ietf-bgp-common {
      revision-date 2023-07-05;
    }
    include ietf-bgp-common-multiprotocol {
      revision-date 2023-07-05;
    }
    include ietf-bgp-common-structure {
      revision-date 2023-07-05;
    }
    include ietf-bgp-neighbor {
      revision-date 2023-07-05;
    }
    include ietf-bgp-rib {
      revision-date 2023-07-05;
    }
    include ietf-bgp-rib-attributes {
      revision-date 2023-07-05;
    }
    include ietf-bgp-rib-tables {
      revision-date 2023-07-05;
    }

    organization
      "IETF IDR Working Group";
    contact
      "WG Web:   <http://datatracker.ietf.org/wg/idr>
       WG List:  <idr@ietf.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com),
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This module describes a YANG model for BGP protocol
       configuration. It is a limited subset of all of the
       configuration parameters available in the variety of vendor
       implementations, hence it is expected that it would be augmented
       with vendor-specific configuration data as needed. Additional
       modules or submodules to handle other aspects of BGP
       configuration, including policy, VRFs, VPNs, and additional
       address families are also expected.

       This model supports the following BGP configuration level
       hierarchy:
```

```
         BGP
           |
           +-> [ global BGP configuration ]
             +-> AFI / SAFI global
           +-> peer group
             +-> [ peer group config ]
             +-> AFI / SAFI [ per-AFI overrides ]
           +-> neighbor
             +-> [ neighbor config ]
             +-> [ optional pointer to peer-group ]
             +-> AFI / SAFI [ per-AFI overrides ]
```

```
      Copyright (c) 2023 IETF Trust and the persons identified as
      authors of the code. All rights reserved.

      Redistribution and use in source and binary forms, with or
      without modification, is permitted pursuant to, and subject to
      the license terms contained in, the Revised BSD License set
      forth in Section 4.c of the IETF Trust's Legal Provisions
      Relating to IETF Documents
      (https://trustee.ietf.org/license-info).

      This version of this YANG module is part of RFC XXXX
      (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
      for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    /*
     * Identity
     */

    identity bgp {
      base rt:routing-protocol;
      description
        "BGP protocol.";
    }
```

```
     /*
      * Groupings
      */

     grouping bgp-errors-common {
       description
         "BGP NOTIFICATION state that is common in the sent and received
          direction.";

       leaf last-notification {
         type yang:date-and-time;
         description
           "The timestamp indicates the time that a BGP
            NOTIFICATION message was last handled.";
         reference
           "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
            4.5.";
       }
       leaf last-error {
         type identityref {
           base bn:bgp-notification;
         }
         description
           "The last NOTIFICATION error for the peer on this
            connection.  If no identity is registered for the
            Error code / Error subcode, this leaf contains the most
            applicable identity for the BGP NOTIFICATION base code.

            The last-error-code and last-error-subcode will always have
            the information received in the NOTIFICATION PDU.";
         reference
           "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
            4.5.";
       }
       leaf last-error-code {
         type uint8;
         description
           "The last NOTIFICATION Error code for the peer on
            this connection.";
         reference
           "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
            4.5.";
       }
       leaf last-error-subcode {
         type uint8;
         description
           "The last NOTIFICATION Error subcode for the peer on
            this connection.";
```

```
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
           4.5.";
      }
    }
    grouping bgp-errors-common-data {
      description
        "BGP NOTIFICATION data state that is common in the sent and
         received direction.";

      leaf last-error-data {
        type binary {
          length "0..65535";
        }
        description
          "Content of the BGP NOTIFICATION PDU's Data field.  This data
           is NOTIFICATION Error code / Error subcode specific.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
           4.5.";
      }
    }

    grouping bgp-encapsulated-errors-common {
      description
        "BGP NOTIFICATION state that is common in the sent and received
         direction that has been encapsulated in a CEASE/HARD RESET
         NOTIFICATION.

         Note that these leaves are only present when carrying the RFC
         8538 encapsulated NOTIFICATION state. In this case,
         the last-error-data leaf will carry the encapsulated Data.";

      leaf last-encapsulated-error {
        type identityref {
          base bn:bgp-notification;
        }
        description
          "The last RFC 8538 encapsulated NOTIFICATION error for the
           peer on this connection.  If no identity is registered for
           the Error code / Error subcode, this leaf contains the most
           applicable identity for the BGP NOTIFICATION base code.

           The last-encapsulated-error-code and
           last-encapsulated-error-subcode will always have the
           encapsulated information received in the CEASE/HARD RESET
           NOTIFICATION PDU's encapsulated ErrCode and Subcode
           fields.";
```

```
        reference
          "RFC 8538: Notification Message Support for BGP Graceful
           Restart, Section 3.1.";
      }
      leaf last-encapsulated-error-code {
        type uint8;
        description
          "The last RFC 8538 encapsulated NOTIFICATION Error code for
           the peer on this connection.";
        reference
          "RFC 8538: Notification Message Support for BGP Graceful
           Restart, Section 3.1.";
      }
      leaf last-encapsulated-error-subcode {
        type uint8;
        description
          "The last RFC 8538 encapsulated NOTIFICATION Error subcode
           for the peer on this connection.";
        reference
          "RFC 8538: Notification Message Support for BGP Graceful
           Restart, Section 3.1.";
      }
    }

    /*
     * Containers
     */

    augment "/rt:routing/rt:control-plane-protocols/"
          + "rt:control-plane-protocol" {
      when "derived-from-or-self(rt:type, 'bgp')" {
        description
          "This augmentation is valid for a routing protocol
           instance of BGP.";
      }
      description
        "BGP protocol augmentation of ietf-routing module
         control-plane-protocol.";
      container bgp {
        description
          "Top-level configuration for the BGP router.";
        container global {
          presence "Enables global configuration of BGP";
          description
            "Global configuration for the BGP router.";
          leaf as {
            type inet:as-number;
            mandatory true;
```

```
            description
              "Local autonomous system number of the router.  Uses
               the 32-bit as-number type from the model in RFC 6991.";
          }
          leaf identifier {
            type yang:dotted-quad;
            description
              "BGP Identifier of the router - an unsigned 32-bit,
               non-zero integer that should be unique within an AS.
               The value of the BGP Identifier for a BGP speaker is
               determined upon startup and is the same for every local
               interface and BGP peer.";
            reference
              "RFC 6286: AS-Wide Unique BGP ID for BGP-4. Section 2.1";
          }
          container distance {
            description
              "Administrative distances (or preferences) assigned to
               routes received from different sources (external, and
               internal).";
            leaf external {
              type uint8 {
                range "1..255";
              }
              description
                "Administrative distances for routes learned from
                 external BGP (eBGP).";
            }
            leaf internal {
              type uint8 {
                range "1..255";
              }
              description
                "Administrative distances for routes learned from
                 internal BGP (iBGP).";
            }
          }
          container confederation {
            description
              "Configuration options specifying parameters when the
               local router is within an autonomous system which is
               part of a BGP confederation.";
            leaf enabled {
              type boolean;
              description
                "When this leaf is set to true it indicates that
                 the local-AS is part of a BGP confederation.";
            }
```

```
        leaf identifier {
          type inet:as-number;
          description
            "Confederation identifier for the autonomous system.";
        }
        leaf-list member-as {
          type inet:as-number;
          description
            "Remote autonomous systems that are to be treated
             as part of the local confederation.";
        }
      }
      container graceful-restart {
        if-feature "bt:graceful-restart";
        description
          "Parameters relating the graceful restart mechanism for
           BGP.";
        reference
          "RFC 4724: Graceful Restart Mechanism for BGP.";
        uses graceful-restart-config;
      }
      uses global-group-use-multiple-paths;
      uses route-selection-options;
      container afi-safis {
        description
          "List of address-families associated with the BGP
           instance.";
        list afi-safi {
          key "name";
          description
            "AFI,SAFI configuration available for the
             neighbor or group.";
          uses mp-afi-safi-config;
          uses state;
          container graceful-restart {
            if-feature "bt:graceful-restart";
            description
              "Parameters relating to BGP graceful-restart";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP.";
            uses mp-afi-safi-graceful-restart-config;
          }
          uses route-selection-options;
          uses structure-add-paths;
          uses global-group-use-multiple-paths;
          uses mp-all-afi-safi-list-contents;
        }
      }
```

```
          uses rt-pol:apply-policy-group;
          uses state;
        }

        container neighbors {
          description
            "Configuration for BGP neighbors.";

          list neighbor {
            key "remote-address";
            description
              "List of BGP neighbors configured on the local system,
               uniquely identified by remote IPv[46] address.";

            leaf remote-address {
              type inet:ip-address;
              description
                "The remote IP address of this entry's BGP peer.";
            }

            leaf peer-group {
              type leafref {
                path "../../../peer-groups/peer-group/name";
              }
              description
                "The peer-group with which this neighbor is
                 associated.";
            }

            leaf local-address {
              type inet:ip-address;
              config false;
              description
                "The local IP address of this entry's BGP connection.";
            }

            leaf local-port {
              type inet:port-number;
              config false;
              description
                "The local port for the TCP connection between
                 the BGP peers.";
            }

            leaf remote-port {
              type inet:port-number;
              config false;
              description
```

```
          "The remote port for the TCP connection
           between the BGP peers.  Note that the
           objects local-addr, local-port, remote-addr, and
           remote-port provide the appropriate
           reference to the standard MIB TCP
           connection table.";
        }

        leaf peer-type {
          type bt:peer-type;
          config false;
          description
            "The type of peering session associated with this
             neighbor.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
                       Section 1.1 for iBGP and eBGP.
             RFC 5065: Autonomous System Configuration
                       for Confederation internal and external.";
        }

        leaf identifier {
          type yang:dotted-quad;
          config false;
          description
            "The BGP Identifier of this entry's BGP peer.
             This entry MUST be 0.0.0.0 unless the
             session state is in the openconfirm or the
             established state.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
                       Section 4.2, 'BGP Identifier'.";
        }

        leaf dynamically-configured {
          type empty;
          config false;
          description
            "When present, this peer has been created
            dynamically.";
        }

        leaf enabled {
          type boolean;
          default "true";
          description
            "Whether the BGP peer is enabled. In cases where the
             enabled leaf is set to false, the local system should
```

```
            not initiate connections to the neighbor, and should
            not respond to TCP connections attempts from the
            neighbor. If the state of the BGP session is
            Established at the time that this leaf is set to
            false, the BGP session should be ceased.

            A transition from 'false' to 'true' will cause
            the BGP Manual Start Event to be generated.
            A transition from 'true' to 'false' will cause
            the BGP Manual Stop Event to be generated.
            This parameter can be used to restart BGP peer
            connections. Care should be used in providing
            write access to this object without adequate
            authentication.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
             Section 8.1.2.";
        }

        uses neighbor-group-config;

        container graceful-restart {
          if-feature "bt:graceful-restart";
          description
            "Parameters relating the graceful restart mechanism for
             BGP";
          reference
            "RFC 4724: Graceful Restart Mechanism for BGP.";
          uses graceful-restart-config;
          leaf peer-restart-time {
            type uint16 {
              range "0..4096";
            }
            config false;
            description
              "The period of time (advertised by the peer) that the
               peer expects a restart of a BGP session to take.";
          }

          leaf peer-restarting {
            type boolean;
            config false;
            description
              "This flag indicates whether the remote neighbor is
               currently in the process of restarting, and hence
               received routes are currently stale.";
          }
```

```
            leaf local-restarting {
              type boolean;
              config false;
              description
                "This flag indicates whether the local neighbor is
                 currently restarting. The flag is cleared after all
                 NLRI have been advertised to the peer, and the
                 End-of-RIB (EOR) marker has been cleared.";
            }

            leaf mode {
              type enumeration {
                enum helper-only {
                  description
                    "The local router is operating in helper-only
                     mode, and hence will not retain forwarding state
                     during a local session restart, but will do so
                     during a restart of the remote peer";
                }
                enum bilateral {
                  description
                    "The local router is operating in both helper
                     mode, and hence retains forwarding state during
                     a remote restart, and also maintains forwarding
                     state during local session restart";
                }
                enum remote-helper {
                  description
                    "The local system is able to retain routes during
                     restart but the remote system is only able to
                     act as a helper";
                }
              }
              config false;
              description
                "This leaf indicates the mode of operation of BGP
                 graceful restart with the peer";
            }
          }

          container prefix-limit {
            description
              "Parameters relating to the prefix limit for the
               AFI-SAFI";

            uses prefix-limit-config-common;

            uses prefix-limit-state-common;
```

```
            }

            container afi-safis {
              description
                "Per-address-family configuration parameters associated
                 with the neighbor";
              uses bgp-neighbor-afi-safi-list;
            }

            leaf session-state {
              type enumeration {
                enum idle {
                  description
                    "Neighbor is down, and in the Idle state of the
                     FSM.";
                }
                enum connect {
                  description
                    "Neighbor is down, and the session is waiting for
                     the underlying transport session to be
                     established.";
                }
                enum active {
                  description
                    "Neighbor is down, and the local system is awaiting
                     a connection from the remote peer.";
                }
                enum opensent {
                  description
                    "Neighbor is in the process of being established.
                     The local system has sent an OPEN message.";
                }
                enum openconfirm {
                  description
                    "Neighbor is in the process of being established.
                     The local system is awaiting a NOTIFICATION or
                     KEEPALIVE message.";
                }
                enum established {
                  description
                    "Neighbor is up - the BGP session with the peer is
                     established.";
                }
              }
              //  notification does not like a non-config statement.
              //  config false;
              description
                "The BGP peer connection state.";
```

```
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 8.1.2.";
    }
    leaf last-established {
      type yang:date-and-time;
      config false;
      description
        "This timestamp indicates the time that the BGP session
         last transitioned in or out of the Established state.
         The value is the timestamp in seconds relative to the
         Unix Epoch (Jan 1, 1970 00:00:00 UTC).

         The BGP session uptime can be computed by clients as
         the difference between this value and the current time
         in UTC (assuming the session is in the Established
         state, per the session-state leaf).";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 8.";
    }

    uses bgp-capabilities;

    container errors {
      config false;
      description
        "Information about BGP NOTIFICATION messages, sent and
         received for this neighbor.";

      container received {
        description
          "BGP NOTIFICATION message state received from this
           neighbor.";
        uses bgp-errors-common;
        uses bgp-encapsulated-errors-common;
        uses bgp-errors-common-data;
      }
      container sent {
        description
          "BGP NOTIFICATION message state sent to this
           neighbor.";
        uses bgp-errors-common;
        uses bgp-encapsulated-errors-common;
        uses bgp-errors-common-data;
      }
    }
```

```
            container statistics {
              config false;
              description
                "Statistics per neighbor.";

              leaf established-transitions {
                type yang:zero-based-counter32;
                description
                  "Number of transitions to the Established state for
                   the neighbor session. This value is analogous to the
                   bgpPeerFsmEstablishedTransitions object from the
                   standard BGP-4 MIB";
                reference
                  "RFC 4273: Definitions of Managed Objects for
                   BGP-4, bgpPeerFsmEstablishedTransitions,
                   RFC 4271: A Border Gateway Protocol 4 (BGP-4),
                   Section 8.";
              }
              container messages {
                description
                  "Counters for BGP messages sent and received from the
                   neighbor";
                leaf total-received {
                  type yang:zero-based-counter32;
                  description
                    "Total number of BGP messages received from this
                     neighbor";
                  reference
                    "RFC 4273: Definitions of Managed Objects for
                     BGP-4, bgpPeerInTotalMessages.";
                }
                leaf total-sent {
                  type yang:zero-based-counter32;
                  description
                    "Total number of BGP messages sent do this
                     neighbor";
                  reference
                    "RFC 4273: Definitions of Managed Objects for
                     BGP-4, bgpPeerOutTotalMessages.";
                }
                leaf updates-received {
                  type yang:zero-based-counter32;
                  description
                    "Number of BGP UPDATE messages received from this
                     neighbor.";
                  reference
                    "RFC 4273: Definitions of Managed Objects for
                     BGP-4, bgpPeerInUpdates.";
```

```
                }
                leaf updates-sent {
                  type yang:zero-based-counter32;
                  description
                    "Number of BGP UPDATE messages sent to this
                     neighbor";
                  reference
                    "RFC 4273: Definitions of Managed Objects for
                     BGP-4, bgpPeerOutUpdates.";
                }
                leaf erroneous-updates-withdrawn {
                  type yang:zero-based-counter32;
                  config false;
                  description
                    "The number of BGP UPDATE messages for which the
                     treat-as-withdraw mechanism has been applied based
                     on erroneous message contents.";
                  reference
                    "RFC 7606: Revised Error Handling for BGP UPDATE
                     Messages, Section 2.";
                }
                leaf erroneous-updates-attribute-discarded {
                  type yang:zero-based-counter32;
                  config false;
                  description
                    "The number of BGP UPDATE messages for which the
                     attribute discard mechanism has been applied based
                     on erroneous message contents.";
                  reference
                    "RFC 7606: Revised Error Handling for BGP UPDATE
                     Messages, Section 2.";
                }
                leaf in-update-elapsed-time {
                  type yang:gauge32;
                  units "seconds";
                  description
                    "Elapsed time (in seconds) since the last BGP
                     UPDATE message was received from the peer.
                     Each time in-updates is incremented,
                     the value of this object is set to zero (0).";
                  reference
                    "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
                     Section 4.3
                     RFC 4271: A Border Gateway Protocol 4 (BGP-4),
                     Established state.";
                }
                leaf notifications-received {
                  type yang:zero-based-counter32;
```

```
            description
              "Number of BGP NOTIFICATION messages indicating an
               error condition has occurred exchanged received
               from this peer.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
               Section 4.5.";
          }
          leaf notifications-sent {
            type yang:zero-based-counter32;
            description
              "Number of BGP NOTIFICATION messages indicating an
               error condition has occurred exchanged sent to
               this peer.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
               Section 4.5.";
          }
          leaf route-refreshes-received {
            type yang:zero-based-counter32;
            description
              "Number of BGP ROUTE-REFRESH messages received from
               this peer.";
            reference
              "RFC 2918: Route Refresh Capability for BGP-4.";
          }
          leaf route-refreshes-sent {
            type yang:zero-based-counter32;
            description
              "Number of BGP ROUTE-REFRESH messages sent to
               this peer.";
            reference
              "RFC 2918: Route Refresh Capability for BGP-4.";
          }
        }
        container queues {
          description
            "Counters related to queued messages associated with
             the BGP neighbor";
          leaf input {
            type yang:gauge32;
            description
              "The number of messages received from the peer
               currently queued";
          }
          leaf output {
            type yang:gauge32;
            description
```

```
                    "The number of messages queued to be sent to the
                     peer";
                }
              }
              action clear {
                if-feature "bt:clear-statistics";
                description
                  "Clear statistics action command.

                   Execution of this command should result in all the
                   counters to be cleared and set to 0.";

                input {
                  leaf clear-at {
                    type yang:date-and-time;
                    description
                      "Time when the clear action needs to be
                       executed.";
                  }
                }
                output {
                  leaf clear-finished-at {
                    type yang:date-and-time;
                    description
                      "Time when the clear action command completed.";
                  }
                }
              }
            }
          }

          notification established {
            leaf remote-address {
              type leafref {
                path "../../neighbor/remote-address";
              }
              description
                "IP address of the neighbor that went into established
                 state.";
            }
            description
              "The established event is generated
               when the BGP FSM enters the established state.";
          }

          notification backward-transition {
            leaf remote-addr {
              type leafref {
```

```
            path "../../neighbor/remote-address";
          }
          description
            "IP address of the neighbor that changed its state from
             established state.";
        }
        container notification-received {
          description
            "If the backwards transition was caused by receiving a
             BGP NOTIFICATION message, this is the information
             received in the NOTIFICATION PDU.";

          uses bgp-errors-common;
          uses bgp-encapsulated-errors-common;
        }
        container notification-sent {
          description
            "If the backwards transition was caused by sending a
             BGP NOTIFICATION message, this is the information
             sent in the NOTIFICATION PDU.";

          uses bgp-errors-common;
          uses bgp-encapsulated-errors-common;
        }
        description
          "The backward-transition event is
           generated when the BGP FSM moves from a higher
           numbered state to a lower numbered state.";
      }
      action clear {
        if-feature "bt:clear-neighbors";
        description
          "Clear neighbors action.";

        input {
          choice operation {
            default operation-admin;
            description
              "The type of operation for the clear action.";
            case operation-admin {
              leaf admin {
                type empty;
                description
                  "Closes the Established BGP session with a BGP
                   NOTIFICATION message with the Administrative
                   Reset error subcode.";
                reference
                  "RFC 4486 - Subcodes for BGP Cease Notification
```

```
                    Message.";
                }
              }
              case operation-hard {
                leaf hard {
                  type empty;
                  description
                    "Closes the Established BGP session with a BGP
                     NOTIFICATION message with the Hard Reset error
                     subcode.";
                  reference
                    "RFC 8538, Section 3 - Notification Message
                     Support for BGP Graceful Restart.";
                }
              }
              case operation-soft {
                leaf soft {
                  type empty;
                  description
                    "Re-sends the current Adj-Rib-Out to this
                     neighbor.";
                }
              }
              case operation-soft-inbound {
                leaf soft-inbound {
                  if-feature "bt:route-refresh";
                  type empty;
                  description
                    "Requests the Adj-Rib-In for this neighbor to be
                     re-sent using the BGP Route Refresh feature.";
                }
              }
            }

            leaf clear-at {
              type yang:date-and-time;
              description
                "Time when the clear action command needs to be
                 executed.";
            }
          }
          output {
            leaf clear-finished-at {
              type yang:date-and-time;
              description
                "Time when the clear action command completed.";
            }
          }
```

```
          }
        }

        container peer-groups {
          description
            "Configuration for BGP peer-groups";

          list peer-group {
            key "name";
            description
              "List of BGP peer-groups configured on the local system -
               uniquely identified by peer-group name";

            leaf name {
              type string;
              description
                "Name of the BGP peer-group";
            }

            uses neighbor-group-config;
            uses structure-dynamic-peers;

            container graceful-restart {
              if-feature "bt:graceful-restart";
              description
                "Parameters relating the graceful restart mechanism for
                 BGP";
              reference
                "RFC 4724: Graceful Restart Mechanism for BGP.";
              uses graceful-restart-config;
            }

            container prefix-limit {
              description
                "Parameters relating to the prefix limit for the
                 AFI-SAFI";

              uses prefix-limit-config-common;
            }

            container afi-safis {
              description
                "Per-address-family configuration parameters
                 associated with the peer-group.";
              list afi-safi {
                key "name";
                description
                  "AFI, SAFI configuration available for the
```

```
                   neighbor or group";
                reference
                  "RFC 4724: Graceful Restart Mechanism for BGP.";
                uses mp-afi-safi-config;
                container graceful-restart {
                  if-feature "bt:graceful-restart";
                  description
                    "Parameters relating to BGP graceful-restart";
                  uses mp-afi-safi-graceful-restart-config;
                }
                uses structure-add-paths;
                uses bgp-neighbor-use-multiple-paths;
                uses mp-all-afi-safi-list-contents;
              }
            }
          }
        }

        uses rib;
      }
    }
  }
  <CODE ENDS>
```

7.1.2. BGP capabilities submodule

```
  <CODE BEGINS> file "ietf-bgp-capabilities@2023-07-05.yang"
  submodule ietf-bgp-capabilities {
    yang-version 1.1;
    belongs-to ietf-bgp {
      prefix bgp;
    }

    import iana-bgp-types {
      prefix bt;
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";

    }
    import iana-routing-types {
      prefix iana-rt-types;
      reference
        "RFC 8294: Common YANG Data Types for the Routing Area.";
    }
    import ietf-inet-types {
      prefix inet;
      reference
        "RFC 6991: Common YANG Data Types.";
```

```
    }

    organization
      "IETF IDR Working Group";
    contact
      "WG Web:   <http://datatracker.ietf.org/wg/idr>
       WG List:  <idr@ietf.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com,
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This sub-module contains groupings that used for BGP
       Capabilities within the BGP module.

       Copyright (c) 2023 IETF Trust and the persons identified as
       authors of the code. All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Revised BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (https://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX
       (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
       for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    grouping bgp-capabilities-common {
      description
        "BGP Capabilities that are used commonly by the
         advertised-capabilities and received-capabilities lists in the
```

```
          bgp-capabilities grouping.";

        leaf code {
          type uint8 {
            range 1..255;
          }
          description
            "BGP Capability Code";
          reference
            "RFC 5492: Capabilities Advertisement with BGP-4, Section
             4.";
        }
        leaf index {
          type uint8;
          description
            "Multiple BGP Capabilities with a given Capability Code may
             be advertised in a BGP OPEN message.  This index permits
             multiple such Capabilities to be represented in the YANG
             model.  It is RECOMMENDED that this index start at one (1)
             and increase by one for each additional instance of the
             Capability.";
          reference
            "RFC 5492: Capabilities Advertisement with BGP-4, Section
             4.";
        }
        leaf name {
          type identityref {
            base bt:bgp-capability;
          }
          description
            "When known, name carries the bgp-capability identity for the
             AFI/SAFI combination as used in the BGP YANG modules.";
        }

        container value {
          description
            "Some BGP Capabilities carry a Capability-specific Capability
             Value.";
          reference
            "RFC 5492: Capabilities Advertisement with BGP-4, Section
             4.";

          container mpbgp {
            when "../../name = 'bt:mp-bgp'";
            description
              "Multi-Protocol BGP-specific values.";
            reference
              "RFC 4760: Multiprotocol Extensions for BGP-4, Section 8.";
```

```
          leaf afi {
            type iana-rt-types:address-family;
            description
             "Address Family Identifier carried in the Multiprotocol
              Extensions Capability.";
            reference
              "RFC 4760: Multiprotocol Extensions for BGP-4, Section
               8.";
          }
          leaf safi {
            type iana-rt-types:bgp-safi;
            description
             "Subsequent Address Family Identifier carried in the
              Multiprotocol Extensions Capability.";
            reference
              "RFC 4760: Multiprotocol Extensions for BGP-4, Section
               8.";
          }
          leaf name {
            type identityref {
              base bt:afi-safi-type;
            }
            description
              "When known, name carries the BGP AFI-SAFI identity for
               the AFI/SAFI combination as used in the BGP YANG
               modules.";
          }
        }

        container graceful-restart {
          when "../../name = 'bt:graceful-restart'";
          description
            "BGP Graceful Restart-specific values.";
          reference
            "RFC 4724: Graceful Restart Mechanism for BGP.";

          leaf flags {
            type bt:graceful-restart-flags;
            description
              "Restart Flags advertised by the Graceful Restart
               Capability";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               3.";
          }

          leaf unknown-flags {
            type bits {
```

```
            bit unknown-2 {
              position 2;
              description
                "Bit 2 was received but is currently RESERVED.";
            }
            bit unknown-3 {
              position 3;
              description
                "Bit 3 was received but is currently RESERVED.";
            }
          }
          description
            "When a bit is exchanged in the Graceful Restart Flags
             field that is unknown to this module, their bit position
             is rendered using the associated unknown bit.";
          reference
            "RFC 4724: Graceful Restart Mechanism for BGP, Section
             3.";
        }

        leaf restart-time {
          type bt:graceful-restart-time-type;
          description
            "The period of time (advertised by the peer) that the
             peer expects a restart of a BGP session to take.";
          reference
            "RFC 4724: Graceful Restart Mechanism for BGP, Section
             3.";
        }

        list afi-safis {
          description
            "List of AFI/SAFIs advertised by the BGP Graceful Restart
             Capability and their AFI/SAFI-specific Flags.";
          leaf afi {
            type iana-rt-types:address-family;
            description
              "Address Family advertised in the BGP Graceful Restart
               Capability.";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               3.";
          }
          leaf safi {
            type iana-rt-types:bgp-safi;
            description
              "Subsequent Address Family advertised in the BGP
               Graceful Restart Capability.";
```

```
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               3.";
          }
          leaf afi-safi-flags {
            type bt:graceful-restart-flags-for-afi;
            description
              "Flags for Address Family advertised per-AFI/SAFI in
               the BGP Graceful Restart Capability.";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               3.";
          }
          leaf afi-safi-unknown-flags {
            type bits {
              bit unknown-1 {
                position 1;
                description
                  "Bit 1 was received but is currently RESERVED.";
              }
              bit unknown-2 {
                position 2;
                description
                  "Bit 2 was received but is currently RESERVED.";
              }
              bit unknown-3 {
                position 3;
                description
                  "Bit 3 was received but is currently RESERVED.";
              }
              bit unknown-4 {
                position 4;
                description
                  "Bit 4 was received but is currently RESERVED.";
              }
              bit unknown-5 {
                position 5;
                description
                  "Bit 5 was received but is currently RESERVED.";
              }
              bit unknown-6 {
                position 6;
                description
                  "Bit 6 was received but is currently RESERVED.";
              }
              bit unknown-7 {
                position 7;
                description
```

```
                    "Bit 7 was received but is currently RESERVED.";
                }
              }
              description
                "When a bit is exchanged in the Graceful Restart Flags
                 for Address Family field that is unknown to this
                 module, their bit position is rendered using the
                 associated unknown bit.";
              reference
                "RFC 4724: Graceful Restart Mechanism for BGP, Section
                 3.";
            }
          }
        }

        container asn32 {
          when "../../name = 'bt:asn32'";
          description
            "Four-byte AS Number-specific values.";
          reference
            "RFC 6793: BGP Support for Four-Octet Autonomous System
             (AS) Number Space, Section 3.";

          leaf as {
            type inet:as-number;
            description
              "Four-byte AS Number carried by the Support for 4-octet
               AS number capability.";
            reference
              "RFC 6793: BGP Support for Four-Octet Autonomous System
               (AS) Number Space, Section 3.";
          }
        }

        container add-paths {
          when "../../name = 'bt:add-paths'";
          description
            "BGP add-paths-specific values.";
          reference
            "RFC 7911: Advertisement of Multiple Paths in BGP.";

          list afi-safis {
            description
              "List of AFI/SAFIs advertised by the BGP ADD-PATH
               Capability and their AFI/SAFI-specific mode.";
            leaf afi {
              type iana-rt-types:address-family;
              description
```

```
            "Address Family Identifier for an instance of the BGP
             ADD-PATH Capability.";
          reference
            "RFC 7911: Advertisement of Multiple Paths in BGP,
             Section 4.";
        }
        leaf safi {
          type iana-rt-types:bgp-safi;
          description
            "Subsequent Address Family Identifier for an instance
             of the BGP ADD-PATH Capability.";
          reference
            "RFC 7911: Advertisement of Multiple Paths in BGP,
             Section 4.";
        }
        leaf mode {
          type enumeration {
            enum receive {
              value 1;
              description
                "The sender of the capability is able to receive
                 multiple paths from its peer.";
              reference
                "RFC 7911: Advertisement of Multiple Paths in BGP,
                 Section 4.";
            }
            enum send {
              value 2;
              description
                "The sender of the capability is able to send
                 multiple paths to its peer.";
              reference
                "RFC 7911: Advertisement of Multiple Paths in BGP,
                 Section 4.";
            }
            enum receive-send {
              value 3;
              description
                "The sender of the capability is able both receive
                 and send multiple paths for its peer.";
              reference
                "RFC 7911: Advertisement of Multiple Paths in BGP,
                 Section 4.";
            }
          }
          description
            "Send/Receive value for a per-AFI/SAFI instance of the
             ADD-PATH Capability.";
```

```
              reference
                "RFC 7911: Advertisement of Multiple Paths in BGP,
                 Section 4.";
            }
          }
        }
      }
    }

    grouping bgp-capabilities {
      description
        "Structural grouping used to include BGP Capabiliites for BGP
         neghbors.";

      container capabilities {
        config false;
        description
          "BGP Capabilities per-neighbor.";
        reference
          "RFC 5492: Capabilities Advertisement with BGP-4.";

        list advertised-capabilities {
          key "code index";
          description
            "List of advertised BGP capabilities, per-peer.
             Identified by the Capability Code and an index.  The
             index covers the case where the same BGP Capability
             may be advertised more than once.";

          uses bgp-capabilities-common;
        }

        list received-capabilities {
          key "code index";
          description
            "List of received BGP capabilities, per-peer.
             Identified by the Capability Code and an index.  The
             index covers the case where the same BGP Capability
             may be advertised more than once.";

          uses bgp-capabilities-common;
        }

        leaf-list negotiated-capabilities {
          type identityref {
            base bt:bgp-capability;
          }
          description
```

```
            "Negotiated BGP capabilities.";
        }
      }
    }
  }
  <CODE ENDS>
```

7.1.3. BGP common submodule

```
  <CODE BEGINS> file "ietf-bgp-common@2023-07-05.yang"
  submodule ietf-bgp-common {
    yang-version 1.1;
    belongs-to ietf-bgp {
      prefix bgp;
    }

    import iana-bgp-types {
      prefix bt;
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }
    import iana-bgp-community-types {
      prefix bct;
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }
    import ietf-interfaces {
      prefix if;
      reference
        "RFC 7223: A YANG Data Model for Interface Management.";
    }
    import ietf-key-chain {
      prefix key-chain;
      reference
        "RFC 8177: YANG Data Model for Key Chains.";
    }
    import ietf-inet-types {
      prefix inet;
      reference
        "RFC 6991: Common YANG Data Types.";
    }
    import ietf-routing-policy {
      prefix rt-pol;
      reference
        "RFC 9067: A YANG Data Model for Routing Policy.";
    }
    import ietf-yang-types {
      prefix yang;
```

```
      reference
        "RFC 6991: Common YANG Data Types.";
    }
    import ietf-bfd-types {
      prefix bfd-types;
      reference
        "RFC 9127: YANG Data Model for Bidirectional Forward
         Detection.";
    }
    import ietf-tcp {
      prefix tcp;
      reference
        "I-D.ietf-tcpm-yang-tcp: Transmission Control Protocol (TCP)
         YANG Model.";
    }
    include ietf-bgp-common-structure {
      revision-date 2023-07-05;
    }

    organization
      "IETF IDR Working Group";
    contact
      "WG Web:   <http://datatracker.ietf.org/wg/idr>
       WG List:  <idr@ietf.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com,
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This sub-module contains common groupings that are common across
       multiple contexts within the BGP module. That is to say that
       they may be application to a subset of global, peer-group, or
       neighbor contexts.

       Copyright (c) 2023 IETF Trust and the persons identified as
       authors of the code. All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Revised BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (https://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX
       (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
```

```
        for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    grouping neighbor-group-timers {
      description
        "Config parameters related to timers associated with the BGP
         peer";
      leaf connect-retry-interval {
        type uint16 {
          range "1..max";
        }
        units "seconds";
        default "120";
        description
          "Time interval (in seconds) for the ConnectRetryTimer. The
           suggested value for this timer is 120 seconds.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
           Section 8.2.2.";
      }
      leaf hold-time {
        type uint16 {
          range "0 | 3..65535";
        }
        units "seconds";
        default "90";
        description
          "Time interval (in seconds) for the HoldTimer established
           with the peer.  When read as operational data (ro), the
           value of this object is calculated by this BGP speaker,
           using the smaller of the values in hold-time that was
           configured (rw) in the running datastore and the Hold Time
           received in the OPEN message.

           This value must be at least three seconds
           if it is not zero (0).
```

```
            If the Hold Timer has not been established
            with the peer this object MUST have a value
            of zero (0).

            If the configured value of hold-time object was
            a value of (0), then when read this object MUST have a
            value of (0) also.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.2,
             RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 10.";
        }
        leaf negotiated-hold-time {
          type uint16;
          config false;
          description
            "The negotiated hold-time for the BGP session";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.2,
             RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 10.";
        }
        leaf keepalive {
          type uint16 {
            range "0..21845";
          }
          units "seconds";
          description
            "When used as a configuration (rw) value, this Time interval
             (in seconds) for the KeepAlive timer configured for this BGP
             speaker with this peer.  A reasonable maximum value for this
             timer would be one-third of the configured hold-time.

             In the absence of explicit configuration of the keepalive
             value, operationally it SHOULD have a value of one-third of
             the negotiated hold-time.

             If the value of this object is zero (0), no periodic
             KEEPALIVE messages are sent to the peer after the BGP
             connection has been established.

             The actual time interval for the KEEPALIVE messages is
             indicated by operational value of keepalive.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.4,
             RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 10.";
        }
        leaf min-as-origination-interval {
          type uint16 {
            range "0..max";
```

```
        }
        units "seconds";
        description
          "Time interval (in seconds) for the MinASOriginationInterval
           timer. The suggested value for this timer is 15 seconds.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
           9.2.1.2,
           RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 10.";
      }
      leaf min-route-advertisement-interval {
        type uint16 {
          range "0..max";
        }
        units "seconds";
        description
          "Time interval (in seconds) for the
           MinRouteAdvertisementInterval timer.
           The suggested value for this timer is 30
           seconds for EBGP connections and 5
           seconds for IBGP connections.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
           9.2.1.1,
           RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 10.";
      }
    }

    grouping bgp-neighbor-use-multiple-paths {
      description
        "Multi-path configuration and state applicable to a BGP
         neighbor";
      container use-multiple-paths {
        description
          "Parameters related to the use of multiple-paths for the same
           NLRI when they are received only from this neighbor";
        leaf enabled {
          type boolean;
          default "false";
          description
            "Whether the use of multiple paths for the same NLRI is
             enabled for the neighbor.";
        }
        container ebgp {
          description
            "Multi-path configuration for eBGP";
          leaf allow-multiple-as {
            type boolean;
```

```
            default "false";
            description
              "Allow multi-path to use paths from different neighboring
               ASes. The default is to only consider multiple paths
               from the same neighboring AS.";
          }
        }
      }
    }

    grouping neighbor-group-config {
      description
        "Neighbor level configuration items.";
      leaf peer-as {
        type inet:as-number;
        description
          "AS number of the peer.";
      }
      leaf local-as {
        type inet:as-number;
        description
          "The local autonomous system number that is to be used when
           establishing sessions with the remote peer or peer group, if
           this differs from the global BGP router autonomous system
           number.";
      }
      leaf remove-private-as {
        type bt:remove-private-as-option;
        description
          "When this leaf is specified, remove private AS numbers from
           updates sent to peers.";
      }
      container route-flap-damping {
        if-feature "bt:damping";
        leaf enable {
          type boolean;
          default "false";
          description
            "Enable route flap damping.";
        }
        leaf suppress-above {
          type decimal64 {
            fraction-digits 1;
          }
          default "3.0";
          description
            "This is the value of the instability metric at which
             route suppression takes place. A route is not installed
```

```
            in the forwarding information base (FIB), or announced
            even if it is reachable during the period that it is
            suppressed.";
        }
        leaf reuse-above {
          type decimal64 {
            fraction-digits 1;
          }
          default "2.0";
          description
            "This is the value of the instability metric at which a
             suppressed route becomes unsuppressed if it is reachable
             but currently suppressed. The value assigned to
             reuse-below must be less than suppress-above.";
        }
        leaf max-flap {
          type decimal64 {
            fraction-digits 1;
          }
          default "16.0";
          description
            "This is the upper limit of the instability metric. This
             value must be greater than the larger of 1 and
             suppress-above.";
        }
        leaf reach-decay {
          type uint32;
          units "seconds";
          default "300";
          description
            "This value specifies the time desired for the instability
             metric value to reach one-half of its current value when
             the route is reachable. This half-life value determines
             the rate at which the metric value is decayed. A smaller
             half-life value makes a suppressed route reusable sooner
             than a larger value.";
        }
        leaf unreach-decay {
          type uint32;
          units "seconds";
          default "900";
          description
            "This value acts the same as reach-decay except that it
             specifies the rate at which the instability metric is
             decayed when a route is unreachable. It should have a
             value greater than or equal to reach-decay.";
        }
        leaf keep-history {
```

```
          type uint32;
          units "seconds";
          default "1800";
          description
            "This value specifies the period over which the route
             flapping history is to be maintained for a given route.
             The size of the configuration arrays described below is
             directly affected by this value.";
        }
        description
          "Routes learned via BGP are subject to weighted route
           dampening.";
      }
      leaf-list send-community {
        if-feature "bct:send-communities";
        type identityref {
          base "bct:send-community-feature";
        }
        description
          "When supported, this tells the router to propagate any
           prefixes that are attached to these community-types.";
      }
      leaf description {
        type string;
        description
          "An optional textual description (intended primarily for use
           with a peer or group";
      }

      container timers {
        description
          "Timers related to a BGP neighbor";
        uses neighbor-group-timers;
      }

      container transport {
        description
          "Transport session parameters for the BGP neighbor";
        uses neighbor-group-transport-config;
      }

      leaf treat-as-withdraw {
        type boolean;
        default "false";
        description
          "Specify whether erroneous UPDATE messages for which the NLRI
           can be extracted are treated as though the NLRI is withdrawn
           - avoiding session reset";
```

```
        reference
          "RFC 7606: Revised Error Handling for BGP UPDATE Messages.";
      }

      container logging-options {
        description
          "Logging options for events related to the BGP neighbor or
           group";
        leaf log-neighbor-state-changes {
          type boolean;
          default "true";
          description
            "Configure logging of peer state changes.  Default is to
             enable logging of peer state changes.

             Note: Documenting demotion from ESTABLISHED state is
                   desirable, but documenting all backward transitions
                   is problematic, and should be avoided.";
        }
      }

      uses structure-neighbor-group-route-reflector;
      uses structure-neighbor-group-as-path-options;
      uses structure-add-paths;
      uses bgp-neighbor-use-multiple-paths;
      uses rt-pol:apply-policy-group;
    }

    grouping neighbor-group-transport-config {
      description
        "Configuration parameters relating to the transport protocol
         used by the BGP session to the peer.";

      leaf local-address {
        type union {
          type inet:ip-address;
          type if:interface-ref;
        }
        description
          "Set the local IP (either IPv4 or IPv6) address to use for
           the session when sending BGP update messages. This may be
           expressed as either an IP address or reference to the name
           of an interface.";
      }

      leaf tcp-mss {
        type tcp:mss;
        description
```

```
          "Maximum Segment Size (MSS) desired on this connection.
           Note, the 'effective send MSS' can be smaller than
           what is configured here.";
         reference
           "RFC 9293: Transmission Control Protocol (TCP)
                      Specification.";
      }

      leaf mtu-discovery {
        type boolean;
        default "true";
        description
          "Turns path mtu discovery for BGP TCP sessions on (true) or
           off (false).";
        reference
          "RFC 1191: Path MTU discovery.";
      }

      container ebgp-multihop {
        description
          "eBGP multi-hop parameters for the BGP peer-group";
        leaf enabled {
          type boolean;
          default "false";
          description
            "When enabled, the referenced group or neighbors are
             permitted to be indirectly connected - including cases
             where the TTL can be decremented between the BGP peers";
        }
        leaf multihop-ttl {
          type uint8;
          description
            "Time-to-live value to use when packets are sent to the
             referenced group or neighbors and ebgp-multihop is
             enabled";
        }
      }

      leaf passive-mode {
        type boolean;
        default "false";
        description
          "Wait for peers to issue requests to open a BGP session,
           rather than initiating sessions from the local router.";
      }

      leaf ttl-security {
        if-feature "bt:ttl-security";
```

```
          type uint8;
          default "255";
          description
            "BGP Time To Live (TTL) security check.";
          reference
            "RFC 5082: The Generalized TTL Security Mechanism (GTSM),
             RFC 7454: BGP Operations and Security.";
        }

        container secure-session {
          must "enabled = 'false' or " +
               "(enabled = 'true' and options)" {
            error-message "When secure-session is enabled " +
              "secure-session options MUST be set";
          }
          description
            "Container for describing how a particular BGP session
             is to be secured.";

          leaf enabled {
            type boolean;
            default "false";
            description
              "When set to true, session is secured with the
               configured options.";
          }

          container options {
            description
              "Options for securing the BGP session.";
            choice option {
              case ao {
                leaf ao-keychain {
                  type key-chain:key-chain-ref;
                  description
                    "Reference to the key chain that will be used by this
                     model. Applicable for TCP-AO and TCP-MD5 only";
                  reference
                    "RFC 8177: YANG Data Model for Key Chains.";
                }
                description
                  "Uses TCP-AO to secure the session. Parameters for
                   those are defined as a grouping in the TCP YANG
                   model.";
                reference
                  "RFC 5925: The TCP Authentication Option.";
              }
```

```
            case md5 {
              leaf md5-keychain {
                type key-chain:key-chain-ref;
                description
                  "Reference to the key chain that will be used by this
                   model. Applicable for TCP-AO and TCP-MD5 only";
                reference
                  "RFC 8177: YANG Data Model for Key Chains.";
              }
              description
                "Uses TCP-MD5 to secure the session. Parameters for
                 those are defined as a grouping in the TCP YANG
                 model.";
              reference
                "RFC 5925: The TCP Authentication Option.";
            }

            case ipsec {
              leaf sa {
                type string;
                description
                  "Security Association (SA) name.";
              }
              description
                "Currently, the IPsec/IKE YANG model has no
                 grouping defined that this model can use. When
                 such a grouping is defined, this model can import
                 the grouping to add the key parameters
                 needed to kick of IKE.";
            }

            description
              "Choice of authentication options.";
          }
        }
      }

      container bfd {
        if-feature "bt:bfd";
        uses bfd-types:client-cfg-parms;
        description
          "BFD client configuration.";
        reference
          "RFC 9127: YANG Data Model for Bidirectional Forwarding
           Detection.";
      }
    }
```

```
    grouping graceful-restart-config {
      description
        "Configuration parameters relating to BGP graceful restart.";
      leaf enabled {
        type boolean;
        default "false";
        description
          "Enable or disable the graceful-restart capability.";
      }
      leaf restart-time {
        type bt:graceful-restart-time-type;
        description
          "Estimated time (in seconds) for the local BGP speaker to
           restart a session. This value is advertise in the graceful
           restart BGP capability.  This is a 12-bit value, referred to
           as Restart Time in RFC4724.  Per RFC4724, the suggested
           default value is <= the hold-time value.";
        reference
          "RFC 4724: Graceful Restart Mechanism for BGP.";
      }
      leaf stale-routes-time {
        type uint32;
        description
          "An upper-bound on the time that stale routes will be
           retained by a router after a session is restarted. If an
           End-of-RIB (EOR) marker is received prior to this timer
           expiring, stale-routes will be flushed upon its receipt - if
           no EOR is received, then when this timer expires stale paths
           will be purged. This timer is referred to as the
           Selection_Deferral_Timer in RFC4724";
        reference
          "RFC 4724: Graceful Restart Mechanism for BGP.";
      }
      leaf helper-only {
        type boolean;
        default "true";
        description
          "Enable graceful-restart in helper mode only. When this leaf
           is set, the local system does not retain forwarding its own
           state during a restart, but supports procedures for the
           receiving speaker, as defined in RFC4724.";
        reference
          "RFC 4724: Graceful Restart Mechanism for BGP.";
      }
    }

    grouping global-group-use-multiple-paths {
      description
```

```
        "Common grouping used for both global and groups which provides
         configuration and state parameters relating to use of multiple
         paths";
      container use-multiple-paths {
        description
          "Parameters related to the use of multiple paths for the
           same NLRI";
        leaf enabled {
          type boolean;
          default "false";
          description
            "Whether the use of multiple paths for the same NLRI is
             enabled for the neighbor. This value is overridden by any
             more specific configuration value.";
        }
        container ebgp {
          description
            "Multi-Path parameters for eBGP";
          leaf allow-multiple-as {
            type boolean;
            default "false";
            description
              "Allow multi-path to use paths from different neighboring
               ASes.  The default is to only consider multiple paths
               from the same neighboring AS.";
          }
          leaf maximum-paths {
            type uint32;
            default "1";
            description
              "Maximum number of parallel paths to consider when using
               BGP multi-path. The default is use a single path.";
          }
        }
        container ibgp {
          description
            "Multi-Path parameters for iBGP";
          leaf maximum-paths {
            type uint32;
            default "1";
            description
              "Maximum number of parallel paths to consider when using
               iBGP multi-path. The default is to use a single path";
          }
        }
      }
    }
```

```
    grouping route-selection-options {
      description
        "Configuration and state relating to route selection options";
      container route-selection-options {
        description
          "Parameters relating to options for route selection";
        leaf always-compare-med {
          type boolean;
          default "false";
          description
            "Compare multi-exit discriminator (MED) value from
             different ASes when selecting the best route.  The default
             behavior is to only compare MEDs for paths received from
             the same AS.";
        }
        leaf ignore-as-path-length {
          type boolean;
          default "false";
          description
            "Ignore the AS path length when selecting the best path.
             The default is to use the AS path length and prefer paths
             with a shorter length.";
        }
        leaf external-compare-router-id {
          type boolean;
          default "true";
          description
            "When comparing similar routes received from external BGP
             peers, use the router-id as a criterion to select the
             active path.";
        }
        leaf advertise-inactive-routes {
          type boolean;
          default "false";
          description
            "Advertise inactive routes to external peers.  The default
             is to only advertise active routes.";
          reference
            "I-D.ietf-idr-best-external: Advertisement of the best
             external route in BGP.";
        }
        leaf enable-aigp {
          type boolean;
          default "false";
          description
            "Flag to enable sending / receiving accumulated IGP
             attribute in routing updates";
          reference
```

```
            "RFC 7311: AIGP Metric Attribute for BGP.";
          }
          leaf ignore-next-hop-igp-metric {
            type boolean;
            default "false";
            description
              "Ignore the IGP metric to the next-hop when calculating BGP
               best-path. The default is to select the route for which
               the metric to the next-hop is lowest";
          }
          leaf enable-med {
            type boolean;
            default "false";
            description
              "Flag to enable sending/receiving of MED metric attribute
               in routing updates.";
          }
          container med-plus-igp {
            leaf enabled {
              type boolean;
              default "false";
              description
                "When enabled allows BGP to use MED and IGP values
                 defined below to determine the optimal route.";
              reference
                "RFC 4451: BGP MED Considerations.";
            }
            leaf igp-multiplier {
              type uint16;
              default 1;
              description
                "Specifies an IGP cost multiplier.";
              reference
                "RFC 4451: BGP MED Considerations.";
            }
            leaf med-multiplier {
              type uint16;
              default 1;
              description
                "Specifies a MED multiplier.";
              reference
                "RFC 4451: BGP MED Considerations.";
            }
            description
              "The med-plus-igp option enables BGP to use the sum of
               MED multiplied by a MED multiplier and IGP cost multiplied
               by IGP cost multiplier to select routes when MED is
               required to determine the optimal route.";
```

```
        }
      }
    }

    grouping state {
      description
        "Grouping containing common counters relating to prefixes and
         paths";

      container statistics {
        config false;
        description
          "Global level statistics.";

        leaf total-paths {
          type yang:gauge32;
          description
            "Total number of BGP paths (BGP routes) within the
             context";
        }

        leaf total-prefixes {
          type yang:gauge32;
          description
            "Total number of BGP prefixes (destinations) received
             within the context";
        }
      }
    }
  }
  <CODE ENDS>
```

7.1.4. BGP common multiprotocol submodule

```
  <CODE BEGINS> file "ietf-bgp-common-multiprotocol@2023-07-05.yang"
  submodule ietf-bgp-common-multiprotocol {
    yang-version 1.1;
    belongs-to ietf-bgp {
      prefix bgp;
    }

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

```
  }
  import ietf-routing-policy {
    prefix rt-pol;
  }

  // meta

  organization
    "IETF IDR Working Group";
  contact
    "WG Web:   <http://datatracker.ietf.org/wg/idr>
     WG List:  <idr@ietf.org>

     Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
              Keyur Patel (keyur at arrcus.com),
              Susan Hares (shares at ndzh.com),
              Jeffrey Haas (jhaas at juniper.net).";

  description
    "This sub-module contains groupings that are related to support
     for multiple protocols in BGP. The groupings are common across
     multiple contexts.

     Copyright (c) 2023 IETF Trust and the persons identified as
     authors of the code. All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject to
     the license terms contained in, the Revised BSD License set
     forth in Section 4.c of the IETF Trust's Legal Provisions
     Relating to IETF Documents
     (https://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC XXXX
     (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
     for full legal notices.

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

  revision 2023-07-05 {
    description
      "Initial Version";
    reference
      "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
```

```
    }

    grouping prefix-limit-config-common {
      description
        "Common configuration for prefix-limit feature.";

      leaf max-prefixes {
        type uint32;
        description
          "Maximum number of prefixes that will be accepted from the
           neighbor";
      }
      leaf warning-threshold-pct {
        type rt-types:percentage;
        description
          "Threshold on number of prefixes that can be received from
           a neighbor before generation of warning messages or log
           entries. Expressed as a percentage of max-prefixes";
      }
      leaf teardown {
        type boolean;
        default false;
        description
          "When 'true', tear down the BGP session when the maximum
           prefix limit is exceeded.  When 'false', only log a
           warning when the maximum prefix limit is exceeded.";
      }
      leaf idle-time {
        type union {
          type uint32;
          type enumeration {
            enum forever {
              description
                "Idle the peer until manually reset.";
            }
          }
        }
        units "seconds";
        description
          "Time interval in seconds after which the BGP session is
           re-established after being torn down due to exceeding the
           max-prefix limit.";
      }
    }

    grouping prefix-limit-state-common {
      description
        "Common operational state for prefix-limit feature.";
```

```
      leaf prefix-limit-exceeded {
        type boolean;
        config false;
        description
          "'true' when the prefix-limit has been exceeded for this
           scope.";
      }
    }


    grouping mp-afi-safi-graceful-restart-config {
      description
        "BGP graceful restart parameters that apply on a per-AFI-SAFI
         basis";
      leaf enabled {
        type boolean;
        must ". = ../../../../graceful-restart/enabled";
        default "false";
        description
          "This leaf indicates whether graceful-restart is enabled for
           this AFI-SAFI.";
        reference
          "RFC 4724: Graceful Restart Mechanism for BGP.";
      }
    }

    grouping mp-afi-safi-config {
      description
        "Configuration parameters used for all BGP AFI-SAFIs";
      leaf name {
        type identityref {
          base bt:afi-safi-type;
        }
        description
          "AFI,SAFI";
      }
      leaf enabled {
        type boolean;
        default "false";
        description
          "This leaf indicates whether this AFI,SAFI is enabled for
           the neighbor or group";
      }
    }

    grouping mp-all-afi-safi-list-contents {
      description
        "A common grouping used for contents of the list that is used
```

```
          for AFI-SAFI entries";
        // import and export policy included for the afi/safi
        uses rt-pol:apply-policy-group;
        container ipv4-unicast {
          when "../name = 'bt:ipv4-unicast'" {
            description
              "Include this container for IPv4 Unicast specific
               configuration";
          }
          description
            "IPv4 unicast configuration options";
          // include common IPv[46] unicast options
          uses mp-ipv4-ipv6-unicast-common;
          // placeholder for IPv4 unicast specific configuration
        }
        container ipv6-unicast {
          when "../name = 'bt:ipv6-unicast'" {
            description
              "Include this container for IPv6 Unicast specific
               configuration";
          }
          description
            "IPv6 unicast configuration options";
          // include common IPv[46] unicast options
          uses mp-ipv4-ipv6-unicast-common;
          // placeholder for IPv6 unicast specific configuration
          // options
        }
        container ipv4-labeled-unicast {
          when "../name = 'bt:ipv4-labeled-unicast'" {
            description
              "Include this container for IPv4 Labeled Unicast specific
               configuration";
          }
          description
            "IPv4 Labeled Unicast configuration options";
          uses mp-all-afi-safi-common;
          // placeholder for IPv4 Labeled Unicast specific config
          // options
        }
        container ipv6-labeled-unicast {
          when "../name = 'bt:ipv6-labeled-unicast'" {
            description
              "Include this container for IPv6 Labeled Unicast specific
               configuration";
          }
          description
            "IPv6 Labeled Unicast configuration options";
```

```
        uses mp-all-afi-safi-common;
        // placeholder for IPv6 Labeled Unicast specific config
        // options.
      }
      container l3vpn-ipv4-unicast {
        when "../name = 'bt:l3vpn-ipv4-unicast'" {
          description
            "Include this container for IPv4 Unicast L3VPN specific
             configuration";
        }
        description
          "Unicast IPv4 L3VPN configuration options";
        // include common L3VPN configuration options
        uses mp-l3vpn-ipv4-ipv6-unicast-common;
        // placeholder for IPv4 Unicast L3VPN specific config options.
      }
      container l3vpn-ipv6-unicast {
        when "../name = 'bt:l3vpn-ipv6-unicast'" {
          description
            "Include this container for unicast IPv6 L3VPN specific
             configuration";
        }
        description
          "Unicast IPv6 L3VPN configuration options";
        // include common L3VPN configuration options
        uses mp-l3vpn-ipv4-ipv6-unicast-common;
        // placeholder for IPv6 Unicast L3VPN specific configuration
        // options
      }
      container l3vpn-ipv4-multicast {
        when "../name = 'bt:l3vpn-ipv4-multicast'" {
          description
            "Include this container for multicast IPv6 L3VPN specific
             configuration";
        }
        description
          "Multicast IPv4 L3VPN configuration options";
        // include common L3VPN multicast options
        uses mp-l3vpn-ipv4-ipv6-multicast-common;
        // placeholder for IPv4 Multicast L3VPN specific configuration
        // options
      }
      container l3vpn-ipv6-multicast {
        when "../name = 'bt:l3vpn-ipv6-multicast'" {
          description
            "Include this container for multicast IPv6 L3VPN specific
             configuration";
        }
```

```
        description
          "Multicast IPv6 L3VPN configuration options";
        // include common L3VPN multicast options
        uses mp-l3vpn-ipv4-ipv6-multicast-common;
        // placeholder for IPv6 Multicast L3VPN specific configuration
        // options
      }
      container l2vpn-vpls {
        when "../name = 'bt:l2vpn-vpls'" {
          description
            "Include this container for BGP-signalled VPLS specific
             configuration";
        }
        description
          "BGP-signalled VPLS configuration options";
        // include common L2VPN options
        uses mp-l2vpn-common;
        // placeholder for BGP-signalled VPLS specific configuration
        // options
      }
      container l2vpn-evpn {
        when "../name = 'bt:l2vpn-evpn'" {
          description
            "Include this container for BGP EVPN specific
             configuration";
        }
        description
          "BGP EVPN configuration options";
        // include common L2VPN options
        uses mp-l2vpn-common;
        // placeholder for BGP EVPN specific configuration options
      }
    }

    // Common groupings across multiple AFI,SAFIs

    grouping mp-all-afi-safi-common {
      description
        "Grouping for configuration common to all AFI,SAFI";
      container prefix-limit {
        description
          "Parameters relating to the prefix limit for the AFI-SAFI";

        uses prefix-limit-config-common;

        uses prefix-limit-state-common;
      }
    }
```

```
  grouping mp-ipv4-ipv6-unicast-common {
    description
      "Common configuration that is applicable for IPv4 and IPv6
       unicast";
    // include common afi-safi options.
    uses mp-all-afi-safi-common;
    // configuration options that are specific to IPv[46] unicast
    leaf send-default-route {
      type boolean;
      default "false";
      description
        "If set to true, send the default-route to the neighbor(s)";
    }
  }

  grouping mp-l3vpn-ipv4-ipv6-unicast-common {
    description
      "Common configuration applied across L3VPN for IPv4
       and IPv6";
    // placeholder -- specific configuration options that are generic
    // across IPv[46] unicast address families.
    uses mp-all-afi-safi-common;
  }

  grouping mp-l3vpn-ipv4-ipv6-multicast-common {
    description
      "Common configuration applied across L3VPN for IPv4
       and IPv6";
    // placeholder -- specific configuration options that are
    // generic across IPv[46] multicast address families.
    uses mp-all-afi-safi-common;
  }

  grouping mp-l2vpn-common {
    description
      "Common configuration applied across L2VPN address
       families";
    // placeholder -- specific configuration options that are
    // generic across L2VPN address families
    uses mp-all-afi-safi-common;
  }

  // Config groupings for common groups

  grouping mp-all-afi-safi-common-prefix-limit-config {
    description
      "Configuration parameters relating to prefix-limits for an
       AFI-SAFI";
```

```
    }
  }
  <CODE ENDS>
```

7.1.5. BGP common structure submodule

```
  <CODE BEGINS> file "ietf-bgp-common-structure@2023-07-05.yang"
  submodule ietf-bgp-common-structure {
    yang-version 1.1;
    belongs-to ietf-bgp {
      prefix bgp;
    }

    import ietf-routing-policy {
      prefix rt-pol;
      reference
        "RFC 9067: A YANG Data Model for Routing Policy Management.";
    }
    import iana-bgp-types {
      prefix bt;
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }
    import ietf-inet-types {
      prefix inet;
      reference
        "RFC 6991: Common YANG Data Types.";
    }
    include ietf-bgp-common-multiprotocol;

    // meta

    organization
      "IETF IDR Working Group";
    contact
      "WG Web:   <http://datatracker.ietf.org/wg/idr>
       WG List:  <idr@ietf.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com),
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This sub-module contains groupings that are common across
       multiple BGP contexts and provide structure around other
       primitive groupings.
```

```
        Copyright (c) 2023 IETF Trust and the persons identified as
        authors of the code. All rights reserved.

        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject to
        the license terms contained in, the Revised BSD License set
        forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    grouping structure-neighbor-group-route-reflector {
      description
        "Structural grouping used to include route reflector
         configuration and state for both BGP neighbors and peer
         groups";
      container route-reflector {
        description
          "Route reflector parameters for the BGP peer-group";
        reference
          "RFC 4456: BGP Route Reflection.";
        leaf cluster-id {
          type bt:rr-cluster-id-type;
          description
            "Route Reflector cluster id to use when local router is
             configured as a route reflector.  Commonly set at the
             group level, but allows a different cluster id to be set
             for each neighbor.";
          reference
            "RFC 4456: BGP Route Reflection: An Alternative to
                       Full Mesh.";
        }
```

```
        leaf client {
          type boolean;
          default "false";
          description
            "Configure the neighbor as a route reflector client.";
          reference
            "RFC 4456: BGP Route Reflection: An Alternative to
                       Full Mesh.";
        }
      }
    }

    grouping structure-neighbor-group-as-path-options {
      description
        "Structural grouping used to include AS_PATH manipulation
         configuration and state for both BGP neighbors and peer
         groups";
      container as-path-options {
        description
          "AS_PATH manipulation parameters for the BGP neighbor or
           group";
        leaf allow-own-as {
          type uint8;
          default "0";
          description
            "Specify the number of occurrences of the local BGP
             speaker's AS that can occur within the AS_PATH before it
             is rejected as looped.";
        }
        leaf replace-peer-as {
          type boolean;
          default "false";
          description
            "Replace occurrences of the peer's AS in the AS_PATH with
             the local autonomous system number";
        }
        leaf disable-peer-as-filter {
          type boolean;
          default "false";
          description
            "When set to true, the system advertises routes to a peer
             even if the peer's AS was in the AS path.  The default
             behavior (false) suppresses advertisements to peers if
             their AS number is in the AS path of the route.";
        }
      }
    }
```

```
    grouping structure-add-paths {
      description
        "Structural grouping used to include ADD-PATHs configuration
         and state.";
      container add-paths {
        if-feature "bt:add-paths";
        description
          "Parameters relating to the advertisement and receipt of
           multiple paths for a single NLRI (add-paths)";
        reference
          "RFC 7911: Advertisements of Multiple Paths in BGP.";
        leaf receive {
          type boolean;
          default "false";
          description
            "Enable ability to receive multiple path advertisements for
             an NLRI from the neighbor or group";
        }
        choice send {
          description
            "Choice of sending the max. number of paths or to send
             all.";
          case max {
            leaf max {
              type uint8;
              description
                "The maximum number of paths to advertise to neighbors
                 for a single NLRI";
            }
          }
          case all {
            leaf all {
              type empty;
              description
                "Send all the path advertisements to neighbors for a
                 single NLRI.";
            }
          }
        }
        leaf eligible-prefix-policy {
          type leafref {
            path "/rt-pol:routing-policy/rt-pol:policy-definitions/"
               + "rt-pol:policy-definition/rt-pol:name";
          }
          description
            "A reference to a routing policy which can be used to
             restrict the prefixes for which add-paths is enabled";
        }
```

```
      }
    }

    grouping structure-dynamic-peers {
      description
        "Structural grouping to contain dyamic BGP peers. Dynamic
         peers are configured from a list of IP prefixes matching the
         IP source address of the dynamic peer.";

      container dynamic-peers {
        description
          "Configuration and operational state for dynamically
           configured peers.";

        list dynamic-peer-list {
          key "prefix";
          description
            "List of peers by IP prefix for this configuration scope
             that are dynamically configured.";

          leaf prefix {
            type inet:ip-prefix;
            description
              "Prefix for dynamic peer at this configuration scope.";
          }
        }
      }
    }
  }
  <CODE ENDS>
```

7.1.6. BGP neighbors submodule

```
  <CODE BEGINS> file "ietf-bgp-neighbor@2023-07-05.yang"
  submodule ietf-bgp-neighbor {
    yang-version 1.1;
    belongs-to ietf-bgp {
      prefix bgp;
    }

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

    import iana-bgp-types {
      prefix bt;
```

```
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    // Include the common submodule

    include ietf-bgp-common;
    include ietf-bgp-common-multiprotocol;
    include ietf-bgp-common-structure;

    // meta

    organization
      "IETF IDR Working Group";
    contact
      "WG Web:   <http://datatracker.ietf.org/wg/idr>
       WG List:  <idr@ietf.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com),
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This sub-module contains groupings that are specific to the
       neighbor context of the BGP module.

       Copyright (c) 2023 IETF Trust and the persons identified as
       authors of the code. All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Revised BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (https://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX
       (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
       for full legal notices.

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

    revision 2023-07-05 {
```

```
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    grouping bgp-neighbor-afi-safi-list {
      description
        "List of address-families associated with the BGP neighbor";
      list afi-safi {
        key "name";
        description
          "AFI, SAFI configuration available for the neighbor or
           group";
        uses mp-afi-safi-config;
        leaf active {
          type boolean;
          config false;
          description
            "This value indicates whether a particular AFI-SAFI has
             been successfully negotiated with the peer. An AFI-SAFI
             may be enabled in the current running configuration, but
             a session restart may be required in order to negotiate
             the new capability.";
        }
        container prefixes {
          config false;
          description
            "Prefix counters for the AFI/SAFI in this BGP session";
          leaf received {
            type yang:zero-based-counter32;
            description
              "The number of prefixes received from the neighbor";
          }
          leaf sent {
            type yang:zero-based-counter32;
            description
              "The number of prefixes advertised to the neighbor";
          }
          leaf installed {
            type yang:gauge32;
            description
              "The number of advertised prefixes installed in the
               Loc-RIB";
          }
        }
        container graceful-restart {
          if-feature "bt:graceful-restart";
```

```
          description
            "Parameters relating to BGP graceful-restart";
          reference
            "RFC 4724: Graceful Restart Mechanism for BGP.";
          uses mp-afi-safi-graceful-restart-config;
          leaf received {
            type boolean;
            config false;
            description
              "This leaf indicates whether the neighbor advertised the
               ability to support graceful-restart for this AFI-SAFI";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               3.";
          }
          leaf advertised {
            type boolean;
            config false;
            description
              "This leaf indicates whether the ability to support
               graceful-restart has been advertised to the peer";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               3.";
          }
          leaf local-forwarding-state-preserved {
            type boolean;
            config false;
            description
              "This leaf indicates whether the local router has
               or would advertise the Forwarding State bit in its
               Graceful Restart capability for this AFI-SAFI.";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               3.";
          }
          leaf forwarding-state-preserved {
            type boolean;
            config false;
            description
              "This leaf indicates whether the neighbor has advertised
               the Forwarding State bit in its Graceful Restart
               capability for this AFI-SAFI.";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               3.";
          }
          leaf end-of-rib-received {
```

```
            type boolean;
            config false;
            description
              "This leaf indicates whether the neighbor has advertised
               the End-of-RIB marker for this AFI-SAFI.";
            reference
              "RFC 4724: Graceful Restart Mechanism for BGP, Section
               2.";
          }
        }
        uses mp-all-afi-safi-list-contents;
        uses bgp-neighbor-use-multiple-paths;
      }
    }
  }
  <CODE ENDS>
```

7.2. BGP notification module

```
  <CODE BEGINS> file "iana-bgp-notification@2023-07-05.yang"
  module iana-bgp-notification {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:iana-bgp-notification";
    prefix bn;

    // meta

    organization
      "IANA";
    contact
      "Internet Assigned Numbers Authority

       Postal: ICANN
               12025 Waterfront Drive, Suite 300
               Los Angeles, CA 90094-2536
               United States of America
       Tel:    +1 310 301 5800
       <mailto:iana@iana.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com),
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This YANG module is maintained by IANA and contains definitions
       of identities used to represent BGP NOTIFICATION Error
       code/Error Subcode tuples.  These values reflect the
```

```
        'BGP Error (Notification) Codes' and 'BGP Error Subcodes'
        registry at IANA.

        Copyright (c) 2023 IETF Trust and the persons identified as
        authors of the code. All rights reserved.

        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject to
        the license terms contained in, the Revised BSD License set
        forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }


    identity bgp-notification {
      description
        "Base identity for BGP NOTIFICATION state.";
    }

    identity message-header-error {
      base bgp-notification;
      description
        "All errors detected while processing the Message Header MUST
         be indicated by sending the NOTIFICATION message with the
         Error Code Message Header Error.

         The value of the 'Message Header Error' Error code is 1.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.1.";
    }
```

```
    identity message-header-unspecific {
      base message-header-error;
      description
        "If no appropriate Error Subcode is defined, then a zero
         (Unspecific) value is used for the Error Subcode field.

         The value of the 'Unspecific Error' subcode is 0.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.5.";
    }

    identity message-header-connection-not-synchronized {
      base message-header-error;
      description
        "If the Marker field of the message header is not as expected,
         then a synchronization error has occurred and the Error
         Subcode MUST be set to Connection Not Synchronized.

         The value of the 'Conection Not Synchronized Error' subcode is
         1.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.1.";
    }

    identity message-header-bad-message-length {
      base message-header-error;
      description
        "If at least one of the following is true:

         - if the Length field of the message header is less than 19 or
           greater than 4096, or

         - if the Length field of an OPEN message is less than the
           minimum length of the OPEN message, or

         - if the Length field of an UPDATE message is less than the
           minimum length of the UPDATE message, or

         - if the Length field of a KEEPALIVE message is not equal to
           19, or

         - if the Length field of a NOTIFICATION message is less than
           the minimum length of the NOTIFICATION message,

         then the Error Subcode MUST be set to Bad Message Length.  The
         Data field MUST contain the erroneous Length field.

         The value of the 'Bad Message Length' Error subcode is 2.";
```

```
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.1.";
    }

    identity message-header-bad-message-type {
      base message-header-error;
      description
        "If the Type field of the message header is not recognized,
         then the Error Subcode MUST be set to Bad Message Type.  The
         Data field MUST contain the erroneous Type field.

         The value of the 'Message Header Error' Error subcode is 3.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.1.";
    }

    identity open-message-error {
      base bgp-notification;
      description
        "All errors detected while processing the OPEN message MUST be
         indicated by sending the NOTIFICATION message with the Error
         Code 'OPEN Message Error'.

         The value of the 'OPEN Message Error' Error code is 2.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.2.";
    }

    identity open-message-unspecific {
      base open-message-error;
      description
        "From Section 4.5:
         If no appropriate Error Subcode is defined, then a zero
         (Unspecific) value is used for the Error Subcode field.

         The value of the 'Unspecific Error' subcode is 0.

         From Section 6.2:
         If one of the Optional Parameters in the OPEN message is
         recognized, but is malformed, then the Error Subcode MUST be
         set to 0 (Unspecific).";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.5.
         RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.2.";
    }

    identity open-unsupported-version-number {
      base open-message-error;
```

```
      description
        "If the version number in the Version field of the received
         OPEN message is not supported, then the Error Subcode MUST be
         set to Unsupported Version Number.  The Data field is a
         2-octet unsigned integer, which indicates the largest,
         locally-supported version number less than the version the
         remote BGP peer bid (as indicated in the received OPEN
         message), or if the smallest, locally-supported version number
         is greater than the version the remote BGP peer bid, then the
         smallest, locally-supported version number.

         The value of the 'Unsupported Version Number' Error subcode is
         1.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.2.";
    }

    identity open-bad-peer-as {
      base open-message-error;
      description
        "If the Autonomous System field of the OPEN message is
         unacceptable, then the Error Subcode MUST be set to Bad Peer
         AS.  The determination of acceptable Autonomous System numbers
         is outside the scope of this protocol.

         The value of the 'Bad Peer AS' Error subcode is 2.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.2.";
    }

    identity open-bad-bgp-id {
      base open-message-error;
      description
        "From RFC 4271, Section 6.2:
         If the BGP Identifier field of the OPEN message is
         syntactically incorrect, then the Error Subcode MUST be set to
         Bad BGP Identifier.  Syntactic correctness means that the BGP
         Identifier field represents a valid unicast IP host address.

         This was updated by RFC 6286:
         For a BGP speaker that supports the AS-wide Unique BGP
         Identifier, the OPEN message error handling related to the BGP
         Identifier is modified as follows:

         If the BGP Identifier field of the OPEN message is zero, or if
         it is the same as the BGP Identifier of the local BGP speaker
         and the message is from an internal peer, then the Error
         Subcode is set to 'Bad BGP Identifier'.
```

```
          The value of the 'Bad BGP Identifier' Error subcode is 3.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.2.
           RFC 6286: Autonomous-System-Wide Unique BGP Identifier for
           BGP-4., Section 2.2.";
      }

      identity open-unsupported-optional-parameter {
        base open-message-error;
        description
          "If one of the Optional Parameters in the OPEN message is not
          recognized, then the Error Subcode MUST be set to Unsupported
          Optional Parameters.

          The value of the 'Unsupported Optional Parameter' Error subcode
          is 4.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.2.";
      }

      identity open-unacceptable-hold-time {
        base open-message-error;
        description
          "If the Hold Time field of the OPEN message is unacceptable,
           then the Error Subcode MUST be set to Unacceptable Hold Time.
           An implementation MUST reject Hold Time values of one or two
           seconds.  An implementation MAY reject any proposed Hold Time.
           An implementation that accepts a Hold Time MUST use the
           negotiated value for the Hold Time.

           The value of the 'Unacceptable Hold Time' Error subcode is
           6.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.2.";
      }

      identity open-unsupported-capability {
        base open-message-error;
        description
          "If a BGP speaker that supports a certain capability determines
           that its peer doesn't support this capability, the speaker MAY
           send a NOTIFICATION message to the peer and terminate peering
           (see Section 'Extensions to Error Handling' for more details).
           For example, a BGP speaker may need to terminate peering if it
           established peering to exchange IPv6 routes and determines
           that its peer does not support Multiprotocol Extensions for
           BGP-4 [RFC4760].  The Error Subcode in the NOTIFICATION
           message is then set to Unsupported Capability.  The message
```

```
          MUST contain the capability or capabilities that cause the
          speaker to send the message.

          The value of the 'Unsupported Capability' Error subcode is
          7.";
        reference
          "RFC 5492: Capabilities Advertisement with BGP-4, Section 3.";
      }

      identity open-role-mismatch {
        base open-message-error;
        description
          "If the BGP Role Capability is advertised, and one is also
           received from the peer, the Roles MUST correspond to the
           relationships in Table 2. If the Roles do not correspond, the
           BGP speaker MUST reject the connection using the Role Mismatch
           Notification (code 2, subcode 11).";
        reference
          "RFC 9234: Route Leak Prevention and Detection Using Roles in
           UPDATE and OPEN Messages, Section 4.2.";
      }

      identity update-message-error {
        base bgp-notification;
        description
          "All errors detected while processing the UPDATE message MUST
           be indicated by sending the NOTIFICATION message with the
           Error Code UPDATE Message Error.

          The value of the 'UPDATE Message Error' Error code is 3.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
      }

      identity update-unspecific {
        base update-message-error;
        description
          "If no appropriate Error Subcode is defined, then a zero
           (Unspecific) value is used for the Error Subcode field.

           The value of the 'Unspecific Error' subcode is 0.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.5.";
      }

      identity update-malformed-attribute-list {
        base update-message-error;
        description
```

```
        "Error checking of an UPDATE message begins by examining the
         path attributes.  If the Withdrawn Routes Length or Total
         Attribute Length is too large (i.e., if Withdrawn Routes
         Length + Total Attribute Length + 23 exceeds the message
         Length), then the Error Subcode MUST be set to Malformed
         Attribute List.

         If any attribute appears more than once in the UPDATE message,
         then the Error Subcode MUST be set to Malformed Attribute
         List.

         The value of the 'Malformed Attribute List' Error subcode is
         1.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
    }

    identity update-unrecognized-well-known-attribute {
      base update-message-error;
      description
        "If any of the well-known mandatory attributes are not
         recognized, then the Error Subcode MUST be set to Unrecognized
         Well-known Attribute.  The Data field MUST contain the
         unrecognized attribute (type, length, and value).

         The value of the 'Unrecognized Well-known Attribute' Error
         subcode is 2.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
    }

    identity update-missing-well-known-attribute {
      base update-message-error;
      description
        "If any of the well-known mandatory attributes are not present,
         then the Error Subcode MUST be set to Missing Well-known
         Attribute.  The Data field MUST contain the Attribute Type
         Code of the missing, well-known attribute.

         The value of the 'Missing Well-known Attribute' Error subcode
         is 3.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
    }

    identity update-attribute-flags-error {
      base update-message-error;
      description
```

```
        "If any recognized attribute has Attribute Flags that conflict
         with the Attribute Type Code, then the Error Subcode MUST be
         set to Attribute Flags Error.  The Data field MUST contain the
         erroneous attribute (type, length, and value).

         The value of the 'Attribute Flags Error' Error subcode is 4.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
    }

    identity update-attribute-length-error {
      base update-message-error;
      description
        "If any recognized attribute has an Attribute Length that
         conflicts with the expected length (based on the attribute
         type code), then the Error Subcode MUST be set to Attribute
         Length Error.  The Data field MUST contain the erroneous
         attribute (type, length, and value).

         The value of the 'Attribute Length Error' Error subcode is
         5.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
    }

    identity update-invalid-origin-attribute {
      base update-message-error;
      description
        "If the ORIGIN attribute has an undefined value, then the Error
         Sub-code MUST be set to Invalid Origin Attribute.  The Data
         field MUST contain the unrecognized attribute (type, length,
         and value).

         The value of the 'Invalid ORIGIN Attribute' Error subcode is
         6.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
    }

    identity update-invalid-next-hop-attribute {
      base update-message-error;
      description
        "If the NEXT_HOP attribute field is syntactically incorrect,
         then the Error Subcode MUST be set to Invalid NEXT_HOP
         Attribute.  The Data field MUST contain the incorrect
         attribute (type, length, and value).  Syntactic correctness
         means that the NEXT_HOP attribute represents a valid IP host
         address.
```

```
          The value of the 'Invalid NEXT_HOP Attribute' Error subcode is
          8.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
      }

      identity open-optional-attribute-error {
        base update-message-error;
        description
          "If an optional attribute is recognized, then the value of this
           attribute MUST be checked.  If an error is detected, the
           attribute MUST be discarded, and the Error Subcode MUST be set
           to Optional Attribute Error.  The Data field MUST contain the
           attribute (type, length, and value).

           The value of the 'Optional Attribute Error' Error subcode is
           9.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
      }

      identity open-invalid-network-field {
        base update-message-error;
        description
          "The NLRI field in the UPDATE message is checked for syntactic
          validity.  If the field is syntactically incorrect, then the
          Error Subcode MUST be set to Invalid Network Field.

          The value of the 'Invalid Network Field' Error subcode is 10.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
      }

      identity open-malformed-as-path {
        base update-message-error;
        description
          "The AS_PATH attribute is checked for syntactic correctness.
           If the path is syntactically incorrect, then the Error Subcode
           MUST be set to Malformed AS_PATH.

           If the UPDATE message is received from an external peer, the
           local system MAY check whether the leftmost (with respect to
           the position of octets in the protocol message) AS in the
           AS_PATH attribute is equal to the autonomous system number of
           the peer that sent the message.  If the check determines this
           is not the case, the Error Subcode MUST be set to Malformed
           AS_PATH.
```

```
          The value of the 'Malformed AS_PATH' Error subcode is 11.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.3.";
      }

      identity hold-timer-expired-error {
        base bgp-notification;
        description
          "If a system does not receive successive KEEPALIVE, UPDATE,
           and/or NOTIFICATION messages within the period specified in
           the Hold Time field of the OPEN message, then the NOTIFICATION
           message with the Hold Timer Expired Error Code is sent and the
           BGP connection is closed.

           The value of the 'Hold Timer Expired' Error code is 4.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.5.";
      }

      identity timer-expired-unspecific {
        base hold-timer-expired-error;
        description
          "If no appropriate Error Subcode is defined, then a zero
           (Unspecific) value is used for the Error Subcode field.

           The value of the 'Unspecific Error' subcode is 0.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.5.";
      }


      identity fsm-error {
        base bgp-notification;
        description
          "Any error detected by the BGP Finite State Machine (e.g.,
           receipt of an unexpected event) is indicated by sending the
           NOTIFICATION message with the Error Code Finite State Machine
           Error.

           The value of the 'Finite State Machine Error' Error code is
           5.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.6.
           RFC 6608: Subcodes for BGP Finite State Machine Error.";
      }

      identity fsm-error-unspecified {
        base fsm-error;
```

```
      description
        "If no appropriate Error Subcode is defined, then a zero
         (Unspecific) value is used for the Error Subcode field.

         The value of the 'Unspecific Error' subcode is 0.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.5.
         RFC 6608: Subcodes for BGP Finite State Machine Error, Section
         3.";
    }

    identity fsm-error-unexpected-in-opensent {
      base fsm-error;
      description
        "If a BGP speaker receives an unexpected message (e.g.,
         KEEPALIVE/ UPDATE/ROUTE-REFRESH message) on a session in
         OpenSent state, it MUST send to the neighbor a NOTIFICATION
         message with the Error Code Finite State Machine Error and the
         Error Subcode 'Receive Unexpected Message in OpenSent State'.
         The Data field is a 1-octet, unsigned integer that indicates
         the type of the unexpected message.

         The value of the 'Receive Unexpected Message in OpenSent
         State' Error subcode is 1.";
      reference
        "RFC 6608: Subcodes for BGP Finite State Machine Error, Section
         4.";
    }

    identity fsm-error-unexpected-in-openconfirm {
      base fsm-error;
      description
        "If a BGP speaker receives an unexpected message (e.g.,
         OPEN/UPDATE/ ROUTE-REFRESH message) on a session in
         OpenConfirm state, it MUST send a NOTIFICATION message with
         the Error Code Finite State Machine Error and the Error
         Subcode 'Receive Unexpected Message in OpenConfirm State' to
         the neighbor.  The Data field is a 1-octet, unsigned integer
         that indicates the type of the unexpected message.

         The value of the 'Receive Unexpected Message in OpenConfirm
         State' Error subcode is 2.";
      reference
        "RFC 6608: Subcodes for BGP Finite State Machine Error, Section
         4.";
    }

    identity fsm-error-unexpected-in-established {
```

```
      base fsm-error;
      description
        "If a BGP speaker receives an unexpected message (e.g., OPEN
         message) on a session in Established State, it MUST send to
         the neighbor a NOTIFICATION message with the Error Code Finite
         State Machine Error and the Error Subcode 'Receive Unexpected
         Message in Established State'.  The Data field is a 1-octet,
         unsigned integer that indicates the type of the unexpected
         message.

         The value of the 'Receive Unexpected Message in Established
         State' Error subcode is 3.";
      reference
        "RFC 6608: Subcodes for BGP Finite State Machine Error, Section
         4.";
    }

    identity cease {
      base bgp-notification;
      description
        "In the absence of any fatal errors (that are indicated in this
         section), a BGP peer MAY choose, at any given time, to close
         its BGP connection by sending the NOTIFICATION message with
         the Error Code Cease.  However, the Cease NOTIFICATION message
         MUST NOT be used when a fatal error indicated by this section
         does exist.

         The value of the 'Cease' Error code is 6.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 6.7.";
    }

    identity cease-max-prefixes {
      base cease;
      description
        "From RFC 4271, Section 6.7:
         A BGP speaker MAY support the ability to impose a
         locally-configured, upper bound on the number of address
         prefixes the speaker is willing to accept from a neighbor.
         When the upper bound is reached, the speaker, under control of
         local configuration, either (a) discards new address prefixes
         from the neighbor (while maintaining the BGP connection with
         the neighbor), or (b) terminates the BGP connection with the
         neighbor.  If the BGP speaker decides to terminate its BGP
         connection with a neighbor because the number of address
         prefixes received from the neighbor exceeds the
         locally-configured, upper bound, then the speaker MUST send
         the neighbor a NOTIFICATION message with the Error Code
```

```
          Cease.

          From RFC 4486:
          If a BGP speaker decides to terminate its peering with a
          neighbor because the number of address prefixes received from
          the neighbor exceeds a locally configured upper bound (as
          described in [BGP-4]), then the speaker MUST send to the
          neighbor a NOTIFICATION message with the Error Code Cease and
          the Error Subcode 'Maximum Number of Prefixes Reached'.  The
          message MAY optionally include the Address Family information
          [BGP-MP] and the upper bound in the 'Data' field, as shown in
          Figure 1, where the meaning and use of the <AFI, SAFI> tuple
          is the same as defined in [BGP-MP], Section 7.

          The value of the 'Maximum Number of Prefixes Reached' Error
          subcode is 1.";
        reference
          "RFC 4486: Subcodes for BGP Cease Notification Message, Section
           4.";
      }

      identity cease-admin-shutdown {
        base cease;
        description
          "If a BGP speaker decides to administratively shut down its
           peering with a neighbor, then the speaker SHOULD send a
           NOTIFICATION message with the Error Code Cease and the Error
           Subcode 'Administrative Shutdown'.

           The value of the 'Administrative Shutdown' Error subcode is
           2.";
        reference
          "RFC 4486: Subcodes for BGP Cease Notification Message, Section
           4.";
      }

      identity cease-peer-deconfigured {
        base cease;
        description
          "If a BGP speaker decides to de-configure a peer, then the
           speaker SHOULD send a NOTIFICATION message with the Error Code
           Cease and the Error Subcode 'Peer De-configured'.

           The value of the 'Peer De-configured' Error subcode is 3.";
        reference
          "RFC 4486: Subcodes for BGP Cease Notification Message, Section
           4.";
      }
```

```
  identity cease-admin-reset {
    base cease;
    description
      "If a BGP speaker decides to administratively reset the peering
       with a neighbor, then the speaker SHOULD send a NOTIFICATION
       message with the Error Code Cease and the Error Subcode
       'Administrative Reset'.

       The value of the 'Administrative Reset' Error subcode is 4.";
     reference
      "RFC 4486: Subcodes for BGP Cease Notification Message, Section
       4.";
  }

  identity cease-connection-rejected {
    base cease;
    description
      "If a BGP speaker decides to disallow a BGP connection (e.g.,
       the peer is not configured locally) after the speaker accepts
       a transport protocol connection, then the BGP speaker SHOULD
       send a NOTIFICATION message with the Error Code Cease and the
       Error Subcode 'Connection Rejected'.

       The value of the 'Connection Rejected' Error subcode is 5.";
    reference
      "RFC 4486: Subcodes for BGP Cease Notification Message, Section
       4.";
  }

  identity cease-other-configuration-change {
    base cease;
    description
      "If a BGP speaker decides to administratively reset the peering
       with a neighbor due to a configuration change other than the
       ones described above, then the speaker SHOULD send a
       NOTIFICATION message with the Error Code Cease and the Error
       Subcode 'Other Configuration Change'.

       The value of the 'Other Configuration Change' Error subcode is
       6.";
    reference
      "RFC 4486: Subcodes for BGP Cease Notification Message, Section
       4.";
  }

  identity cease-connection-collision {
    base cease;
    description
```

```
        "If a BGP speaker decides to send a NOTIFICATION message with
         the Error Code Cease as a result of the collision resolution
         procedure (as described in [BGP-4]), then the subcode SHOULD
         be set to 'Connection Collision Resolution'.

         The value of the 'Connection Collision Resolution' Error
         subcode is 7.";
      reference
        "RFC 4486: Subcodes for BGP Cease Notification Message, Section
         4.";
    }

    identity cease-out-of-resources {
      base cease;
      description
        "If a BGP speaker runs out of resources (e.g., memory) and
         decides to reset a session, then the speaker MAY send a
         NOTIFICATION message with the Error Code Cease and the Error
         Subcode 'Out of Resources'.

         The value of the 'Out of Resources' Error subcode is 8.";
      reference
        "RFC 4486: Subcodes for BGP Cease Notification Message, Section
         4.";
    }

    identity cease-hard-reset {
      base cease;
      description
        "[RFC 8538] defines a new subcode for BGP Cease NOTIFICATION
         messages, called the Hard Reset subcode.  The value of this
         subcode is [9].  In this document, a BGP Cease NOTIFICATION
         message with the Hard Reset subcode is referred to as a 'Hard
         Reset message' or simply as a 'Hard Reset'.

         The value of the 'Hard Reset' Error subcode is 9.";
      reference
        "RFC 8538: Notification Message Support for BGP Graceful
         Restart, Section 3.";
    }

    identity cease-bfd-down {
      base cease;
      description
        "When a BGP session is terminated due to a BFD session going
         into the Down state, the BGP Speaker SHOULD send a
         NOTIFICATION message with the Error Code Cease and the Error
         Subcode 'BFD Down'.
```

```
          The value of the 'BFD Down' Error subocde is 10.";
      reference
        "draft-ietf-idr-bfd-subcode-05: A BGP Cease Notification
        Subcode For Bidirectional Forwarding Detection (BFD), Section
        2.";
    }

    identity route-refresh-message-error {
      base bgp-notification;
      description
        "The ROUTE-REFRESH Message Error code was defined in RFC 7313.

        The value of the 'ROUTE-REFRESH Message Error' Error code is
        7.";
      reference
        "RFC 7313: Enhanced Route Refresh Capability for BGP-4, Section
         5.";
    }

    identity route-refresh-reserved {
      base route-refresh-message-error;
      description
        "If no appropriate Error Subcode is defined, then a zero
         (Unspecific) value is used for the Error Subcode field.

         The value of the 'Unspecific Error' subcode is 0.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.5.
         RFC 7313: Enhanced Route Refresh Capability for BGP-4, Section
         6.";
    }

    identity route-refresh-invalid-message-length {
      base route-refresh-message-error;
      description
        "If the length, excluding the fixed-size message header, of the
         received ROUTE-REFRESH message with Message Subtype 1 and 2 is
         not 4, then the BGP speaker MUST send a NOTIFICATION message
         with the Error Code of 'ROUTE-REFRESH Message Error' and the
         subcode of 'Invalid Message Length'.  The Data field of the
         NOTIFICATION message MUST contain the complete ROUTE-REFRESH
         message.

         The value of the 'Invalid Message Length' Error subcode is
         1.";
      reference
        "RFC 7313: Enhanced Route Refresh Capability for BGP-4, Section
         5.";
```

```
    }
  }
  <CODE ENDS>
```

7.3. BGP types module

```
  <CODE BEGINS> file "iana-bgp-types@2023-07-05.yang"
  module iana-bgp-types {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:iana-bgp-types";
    prefix bt;

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

    // meta

    organization
      "IANA";
    contact
      "Internet Assigned Numbers Authority

       Postal: ICANN
               12025 Waterfront Drive, Suite 300
               Los Angeles, CA 90094-2536
               United States of America
       Tel:    +1 310 301 5800
       <mailto:iana@iana.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com),
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This module contains general data definitions for use in BGP.
       It can be imported by modules that make use of BGP attributes.

       Copyright (c) 2023 IETF Trust and the persons identified as
       authors of the code. All rights reserved.
```

```
        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject to
        the license terms contained in, the Revised BSD License set
        forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG module for Border Gateway Protocol (BGP-4).";
    }

    /*
     * Features.
     */

    feature add-paths {
      description
        "Advertisement of multiple paths for the same address prefix
         without the new paths implicitly replacing any previous
         ones.";
      reference
        "RFC 7911: Advertisement of Multiple Paths in BGP.";
    }

    feature bfd {
      description
        "Support for BFD detection of BGP neighbor reachability.";
      reference
        "RFC 5880: Bidirectional Forward Detection (BFD),
         RFC 5881: Bidirectional Forward Detection for IPv4 and IPv6
                   (Single Hop),
         RFC 5883: Bidirectional Forwarding Detection (BFD) for
                   Multihop Paths.";
    }
```

```
  feature clear-neighbors {
    description
      "Clearing of BGP neighbors is supported.";
  }

  feature clear-routes {
    description
      "Clearing of BGP routes is supported.";
  }

  feature clear-statistics {
    description
      "Clearing of BGP statistics is supported.";
  }

  feature damping {
    description
      "Weighted route dampening is supported.";
  }

  feature graceful-restart {
    description
      "Graceful restart as defined in RFC 4724 is supported.";
  }

  feature route-refresh {
    description
      "Support for the BGP Route Refresh capability.";
    reference
      "RFC 2918: Route Refresh Capability for BGP-4.";
  }

  feature ttl-security {
    description
      "BGP Time To Live (TTL) security check support.";
    reference
      "RFC 5082: The Generalized TTL Security Mechanism (GTSM).";
  }

  /*
   * Identities.
   */

  /* BGP Origin Attribute Types. */

  typedef bgp-origin-attr-type {
    type enumeration {
      enum igp {
```

```
          description
            "Origin of the NLRI is internal";
        }
        enum egp {
          description
            "Origin of the NLRI is EGP";
        }
        enum incomplete {
          description
            "Origin of the NLRI is neither IGP or EGP";
        }
      }
      description
        "Type definition for standard BGP origin attribute";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Sec 4.3.";
    }

    /* BGP AS_PATH Segment Type Identities. */

    identity as-path-segment-type {
      description
        "Base AS Path Segment Type. In [BGP-4], the path segment type
         is a 1-octet field with the following values defined.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.3.";
    }

    identity as-set {
      base as-path-segment-type;
      description
        "Unordered set of autonomous systems that a route in the UPDATE
         message has traversed.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.3.";
    }

    identity as-sequence {
      base as-path-segment-type;
      description
        "Ordered set of autonomous systems that a route in the UPDATE
         message has traversed.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 4.3.";
    }

    identity as-confed-sequence {
      base as-path-segment-type;
```

```
      description
        "Ordered set of Member Autonomous Systems in the local
         confederation that the UPDATE message has traversed.";
      reference
        "RFC 5065: Autonomous System Configuration for BGP.";
    }

    identity as-confed-set {
      base as-path-segment-type;
      description
        "Unordered set of Member Autonomous Systems in the local
         confederation that the UPDATE message has traversed.";
      reference
        "RFC 5065: Autonomous System Configuration for BGP.";
    }

    /* BGP Peer-Types */

    typedef peer-type {
      type enumeration {
        enum internal {
          description
            "Internal (IBGP) peer";
        }
        enum external {
          description
            "External (EBGP) peer";
        }
        enum confederation-internal {
          description
            "Confederation Internal (IBGP) peer.";
        }
        enum confederation-external {
          description
            "Confederation External (EBGP) peer.";
        }
      }
      description
        "Labels a peer or peer group as explicitly internal,
         external, or the related confederation type.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Sec 1.1.
         RFC 5065: Autonomous System Configuration for BGP.";
    }

    /* BGP Capability Identities. */

    identity bgp-capability {
```

```
    description
      "Base identity for a BGP capability";
  }

  identity mp-bgp {
    base bgp-capability;
    description
      "Multi-protocol extensions to BGP";
    reference
      "RFC 4760: Multiprotocol Extentions for BGP-4.";
  }

  identity route-refresh {
    base bgp-capability;
    description
      "The BGP route-refresh functionality";
    reference
      "RFC 2918: Route Refresh Capability for BGP-4.";
  }

  identity asn32 {
    base bgp-capability;
    description
      "4-byte (32-bit) AS number functionality";
    reference
      "RFC6793: BGP Support for Four-Octet Autonomous System (AS)
                Number Space.";
  }

  identity graceful-restart {
    if-feature "graceful-restart";
    base bgp-capability;
    description
      "Graceful restart functionality";
    reference
      "RFC 4724: Graceful Restart Mechanism for BGP.";
  }

  identity add-paths {
    if-feature "add-paths";
    base bgp-capability;
    description
      "Advertisement of multiple paths for the same address prefix
       without the new paths implicitly replacing any previous
       ones.";
    reference
      "RFC 7911: Advertisement of Multiple Paths in BGP.";
  }
```

```
    /* BGP AFI-SAFI Type Identities. */

    identity afi-safi-type {
      description
        "Base identity type for AFI,SAFI tuples for BGP-4";
      reference
        "RFC4760: Multiprotocol Extentions for BGP-4.";
    }

    identity ipv4-unicast {
      base afi-safi-type;
      description
        "IPv4 unicast (AFI,SAFI = 1,1)";
      reference
        "RFC4760: Multiprotocol Extentions for BGP-4.";
    }

    identity ipv4-labeled-unicast {
      base afi-safi-type;
      description
        "Labeled IPv4 unicast (AFI,SAFI = 1,4)";
      reference
        "RFC 8277: Using BGP to Bind MPLS Labels to Address Prefixes.";
    }

    identity ipv6-unicast {
      base afi-safi-type;
      description
        "IPv6 unicast (AFI,SAFI = 2,1)";
      reference
        "RFC4760: Multiprotocol Extentions for BGP-4.";
    }

    identity ipv6-labeled-unicast {
      base afi-safi-type;
      description
        "Labeled IPv6 unicast (AFI,SAFI = 2,4)";
      reference
        "RFC 8277: Using BGP to Bind MPLS Labels to Address Prefixes.";
    }

    identity l3vpn-ipv4-unicast {
      base afi-safi-type;
      description
        "Unicast IPv4 MPLS L3VPN (AFI,SAFI = 1,128)";
      reference
        "RFC 4364: BGP/MPLS IP Virtual Private Networks (VPNs).";
    }
```

```
identity l3vpn-ipv4-multicast {
  base afi-safi-type;
  description
    "Multicast IPv4 MPLS L3VPN (AFI,SAFI = 1,129)";
  reference
    "RFC 6514: BGP Encodings and Procedures for Multicast in
     MPLS/BGP IP VPNs.";
}

identity l3vpn-ipv6-unicast {
  base afi-safi-type;
  description
    "Unicast IPv6 MPLS L3VPN (AFI,SAFI = 2,128)";
  reference
    "RFC 4659: BGP-MPLS IP Virtual Private Network (VPN) Extension
     for IPv6 VPN.";
}

identity l3vpn-ipv6-multicast {
  base afi-safi-type;
  description
    "Multicast IPv6 MPLS L3VPN (AFI,SAFI = 2,129)";
  reference
    "RFC 6514: BGP Encodings and Procedures for Multicast in
     MPLS/BGP IP VPNs.";
}

identity l2vpn-evpn {
  base afi-safi-type;
  description
    "BGP MPLS Based Ethernet VPN (AFI,SAFI = 25,70)";
  reference
    "RFC 7432: BGP MPLS-Based Ethernet VPN.";
}

identity l2vpn-vpls {
  base afi-safi-type;
  description
    "BGP-signalled VPLS (AFI,SAFI = 25,65)";
  reference
    "RFC 4761: Virtual Private LAN Service (VPLS) Using BGP for
     Auto-Discovery and Signaling.";
}

/* BGP Remove Private AS Identities. */

identity remove-private-as-option {
  description
```

```
        "Base identity for options for removing private autonomous
         system numbers from the AS_PATH attribute";
    }

    identity private-as-remove-all {
      base remove-private-as-option;
      description
        "Strip all private autonomous system numbers from the AS_PATH.
         This action is performed regardless of the other content of
         the AS_PATH attribute, and for all instances of private AS
         numbers within that attribute.";
    }

    identity private-as-replace-all {
      base remove-private-as-option;
      description
        "Replace all instances of private autonomous system numbers in
         the AS_PATH with the local BGP speaker's autonomous system
         number. This action is performed regardless of the other
         content of the AS_PATH attribute, and for all instances of
         private AS number within that attribute.";
    }

    /*
     * Typedefs.
     */

    typedef bgp-session-direction {
      type enumeration {
        enum inbound {
          description
            "Refers to all NLRI received from the BGP peer";
        }
        enum oubound {
          description
            "Refers to all NLRI advertised to the BGP peer";
        }
      }
      description
        "Type to describe the direction of NLRI transmission";
    }

    /* BGP Route Reflector Types. */

    typedef rr-cluster-id-type {
      type union {
        type uint32;
        type yang:dotted-quad;
```

```
      }
      description
        "Union type for route reflector cluster ids:
         option 1: 4-byte number
         option 2: IP address";
      reference
        "RFC 4456: BGP Route Reflection.";
    }

    /* BGP Graceful Restart Types. */

    typedef graceful-restart-time-type {
      type uint16 {
        range "0..4096";
      }
      units "seconds";
      description
        "This is the estimated time (in seconds) it will take for the
         BGP session to be re-established after a restart.  This can be
         used to speed up routing convergence by its peer in case that
         the BGP speaker does not come back after a restart.";
      reference
         "RFC 4724: Graceful Restart Mechanism for BGP, Section 3.";
    }

    typedef graceful-restart-flags {
      type bits {
        bit restart {
          position 0;
          description
            "The most significant bit is defined as the Restart
             State (R) bit, which can be used to avoid possible
             deadlock caused by waiting for the End-of-RIB marker
             when multiple BGP speakers peering with each other
             restart.  When set (value 1), this bit indicates
             that the BGP speaker has restarted, and its peer
             MUST NOT wait for the End-of-RIB marker from the
             speaker before advertising routing information to
             the speaker.";
          reference
            "RFC 4724: Graceful Restart Mechanism for BGP,
             Section 3.";
        }
        bit notification {
          position 1;
          description
            "The second most significant bit is defined in [RFC 8538]
             as the Graceful Notification ('N') bit.  It is used to
```

```
            indicate Graceful Restart support for BGP NOTIFICATION
            messages.  A BGP speaker indicates support for the
            procedures in this document by advertising a Graceful
            Restart Capability with its 'N' bit set (value 1).";
          reference
            "RFC 8538: Notification Message Support for BGP Graceful
             Restart, Section 2.";
        }
      }
      description
        "Restart Flags bits as defined for BGP Graceful Restart and its
         extensions.";
      reference
        "RFC 4724: Graceful Restart Mechanism for BGP,
         RFC 8538: Notification Message Support for BGP Graceful
         Restart.";
    }

    typedef graceful-restart-flags-for-afi {
      type bits {
        bit forwarding-preserved {
          position 0;
          description
            "The most significant bit is defined as the
             Forwarding State (F) bit, which can be used to
             indicate whether the forwarding state for routes
             that were advertised with the given AFI and SAFI
             has indeed been preserved during the previous BGP
             restart.  When set (value 1), the bit indicates
             that the forwarding state has been preserved.";
          reference
            "RFC 4724: Graceful Restart Mechanism for BGP,
             Section 3.";
        }
      }
      description
        "Flags for Address Family bits as defined for BGP Graceful
         Restart and its extensions.";
      reference
        "RFC 4724: Graceful Restart Mechanism for BGP.";
    }

    /* BGP Remove Private AS Types. */

    typedef remove-private-as-option {
      type identityref {
        base remove-private-as-option;
      }
```

```
      description
        "Set of options for configuring how private AS path numbers
         are removed from advertisements";
    }

  }
  <CODE ENDS>
```

7.4. BGP community types module

```
  <CODE BEGINS> file "iana-bgp-community-types@2023-07-05.yang"
  module iana-bgp-community-types {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:iana-bgp-community-types";
    prefix bct;

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

    // meta

    organization
      "IANA";
    contact
      "Internet Assigned Numbers Authority

       Postal: ICANN
               12025 Waterfront Drive, Suite 300
               Los Angeles, CA 90094-2536
               United States of America
       Tel:    +1 310 301 5800
       <mailto:iana@iana.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com),
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This module contains data definitions for BGP Communities in
       their various forms.  It can be imported by modules that make
       use of BGP attributes.
```

```
        This YANG module is maintained by IANA and reflects the
        'BGP Identities for Community' and 'BGP definitions for
        Community type' registries.

        Copyright (c) 2023 IETF Trust and the persons identified as
        authors of the code. All rights reserved.

        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject to
        the license terms contained in, the Revised BSD License set
        forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG module for Border Gateway Protocol (BGP-4).";
    }

    /*
     * Features.
     */

    feature send-communities {
      description
        "Enable the propagation of communities.";
    }

    /*
     * Identities.
     */

    /* BGP Well-Known Standard (RFC 1997) Community Identities. */

    identity bgp-well-known-std-community {
      description
```

```
        "Base identity for reserved communities within the standard
         community space defined by RFC 1997. These communities must
         fall within the range 0xFFFF0000 to 0xFFFFFFFF";
      reference
        "RFC 1997: BGP Communities Attribute.";
    }

    identity no-export {
      base bgp-well-known-std-community;
      description
        "Do not export NLRI received carrying this community outside
         the bounds of this autonomous system, or this confederation
         (if the local autonomous system is a confederation member AS).
          This community has a value of 0xFFFFFF01.";
      reference
        "RFC 1997: BGP Communities Attribute.";
    }

    identity no-advertise {
      base bgp-well-known-std-community;
      description
        "All NLRI received carrying this community must not be
         advertised to other BGP peers. This community has a value of
         0xFFFFFF02.";
      reference
        "RFC 1997: BGP Communities Attribute.";
    }

    identity no-export-subconfed {
      base bgp-well-known-std-community;
      description
        "All NLRI received carrying this community must not be
         advertised to external BGP peers - including over
         confederation sub-AS boundaries. This community has a value of
         0xFFFFFF03.";
      reference
        "RFC 1997: BGP Communities Attribute.";
    }

    identity no-peer {
      base bgp-well-known-std-community;
      description
        "An autonomous system receiving NLRI tagged with this community
         is advised not to re-advertise the NLRI to external bilateral
         peer autonomous systems. An AS may also filter received NLRI
         from bilateral peer sessions when they are tagged with this
         community value. This community has a value of 0xFFFFFF04.";
      reference
```

```
        "RFC 3765: NOPEER Community for BGP.";
    }

    /* BGP Community Send Identities. */

    identity send-community-feature {
      description
        "Base identity to identify send-community feature.";
    }

    identity standard {
      base send-community-feature;
      description
        "Send standard communities.";
      reference
        "RFC 1997: BGP Communities Attribute.";
    }

    identity extended {
      base send-community-feature;
      description
        "Send extended communities.";
      reference
        "RFC 4360: BGP Extended Communities Attribute.";
    }

    identity large {
      base send-community-feature;
      description
        "Send large communities.";
      reference
        "RFC 8092: BGP Large Communities Attribute.";
    }

    /*
     * Typedefs.
     */

    /* BGP Community Types. */

    typedef bgp-community-regexp-type {
      type string;
      description
        "Type definition for communities specified as regular
         expression patterns.

         A compliant implementation of this type MUST accept a POSIX.2
         Extended Regular Expression excluding the following features:
```

```
          - character class expressions (Section 9.3.5)
          - collating symbols (Section 9.3.5)
          - equivalence classes (Section 9.3.5)
          - back-reference expressions (Section 9.3.6)

          Implementations MAY accept additional forms of regular
          expressions as long as the minimally compliant form documented
          above is accepted.";
      reference
        "IEEE Std 1003.1-2017: The Open Group Base Specifications Issue
         7, 2018 edition.";
    }

    typedef bgp-std-community-type {
      type union {
        type uint32;
        type string {
          pattern '([0-9]|[1-9][0-9]{1,3}|[1-5][0-9]{4}|'
                + '6[0-5][0-9]{3}|66[0-4][0-9]{2}|'
                + '665[0-2][0-9]|6653[0-5]):'
                + '([0-9]|[1-9][0-9]{1,3}|[1-5][0-9]{4}|'
                + '6[0-5][0-9]{3}|66[0-4][0-9]{2}|'
                + '665[0-2][0-9]|6653[0-5])';
        }
      }
      description
        "Type definition for standard community attributes.";
      reference
        "RFC 1997: BGP Communities Attribute.";
    }

    typedef bgp-well-known-community-type {
      type identityref {
        base bgp-well-known-std-community;
      }
      description
        "Type definition for well-known IETF community attribute
         values.";
      reference
        "IANA Border Gateway Protocol (BGP) Well Known Communities";
    }

    /* BGP Large Community Types. */

    typedef bgp-large-community-type {
      type string {
        // 4-octets global:4-octets local part-1:4-octets local part-2.
        pattern '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
```

```
              + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[0-9]):'
              + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
              + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[0-9]):'
              + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
              + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[0-9])';
      }
      description
        "Type definition for a large BGP community";
      reference
        "RFC 8092: BGP Large Communities Attribute.";
    }

    /* IPv6 BGP Exteneded Community Types. */

    typedef bgp-ipv6-ext-community-type {
      type union {
        type string {
          pattern 'ipv6-route\-target:'
                + '((:|[0-9a-fA-F]{0,4}):)([0-9a-fA-F]{0,4}:){0,5}'
                + '((([0-9a-fA-F]{0,4}:)?(:|[0-9a-fA-F]{0,4}))|'
                + '(((25[0-5]|2[0-4][0-9]|1[0-9]{2}|[1-9]?[0-9])\.){3}'
                + '(25[0-5]|2[0-4][0-9]|1[0-9]{2}|[1-9]?[0-9])))'
                + ':'
                + '(6553[0-5]|655[0-2][0-9]|65[0-4][0-9]{2}|'
                + '6[0-4][0-9]{3}|'
                + '[1-5][0-9]{4}|[1-9][0-9]{0,3}|0)';
           pattern 'ipv6-route\-target:'
                + '((([^:]+:){6}(([^:]+:[^:]+)|(.*\..*)))|'
                + '((([^:]+:)*[^:]+)?::(([^:]+:)*[^:]+)?))'
                + ':'
                + '(6553[0-5]|655[0-2][0-9]|65[0-4][0-9]{2}|'
                + '6[0-4][0-9]{3}|'
                + '[1-5][0-9]{4}|[1-9][0-9]{0,3}|0)';
          /*
           * description
           *   "Type 0x00, Sub-Type 0x02: IPv6 Route-Target
           *    ipv6-route-target:(IPv6 address):(local-admin)
           *    16 octets global administrator and 2 octets local
           *    administrator.
           *
           *    Note: The patterns above are excerpted from RFC 8294.";
           * reference
           *   "RFC 5701: IPv6 Address Specific BGP Extended Community
           *    Attribute, Section 3.";
           */
        }

        type string {
```

```
          pattern 'ipv6-route\-origin:'
               + '((:|[0-9a-fA-F]{0,4}):)([0-9a-fA-F]{0,4}:){0,5}'
               + '((([0-9a-fA-F]{0,4}:)?(:|[0-9a-fA-F]{0,4}))|'
               + '(((25[0-5]|2[0-4][0-9]|1[0-9]{2}|[1-9]?[0-9])\.){3}'
               + '(25[0-5]|2[0-4][0-9]|1[0-9]{2}|[1-9]?[0-9])))'
               + ':'
               + '(6553[0-5]|655[0-2][0-9]|65[0-4][0-9]{2}|'
               + '6[0-4][0-9]{3}|'
               + '[1-5][0-9]{4}|[1-9][0-9]{0,3}|0)';
           pattern 'ipv6-route\-origin:'
               + '((([^:]+:){6}(([^:]+:[^:]+)|(.*\..*)))|'
               + '((([^:]+:)*[^:]+)?::(([^:]+:)*[^:]+)?))'
               + ':'
               + '(6553[0-5]|655[0-2][0-9]|65[0-4][0-9]{2}|'
               + '6[0-4][0-9]{3}|'
               + '[1-5][0-9]{4}|[1-9][0-9]{0,3}|0)';
          /*
           * description
           *   "Type 0x00, Sub-Type 0x03: IPv6 Route-Origin
           *    ipv6-route-origin:(IPv6 address):(local-admin)
           *    16 octets global administrator and 2 octets local
           *    administrator.
           *
           *    Note: The patterns above are excerpted from RFC 8294.";
           * reference
           *   "RFC 5701: IPv6 Address Specific BGP Extended Community
           *    Attribute, Section 3.";
           */
        }

        type string {
          // raw with 20 octets
          pattern 'ipv6-raw:'
               + '([0-9A-Fa-f][0-9A-Fa-f]:){19}'
               + '[0-9A-Fa-f][0-9A-Fa-f]';
        }
      }
      description
        "Type definition for IPv6 extended community attributes.
         It includes a way to specify a 'raw' string that
         is followed by 20 bytes of octet string to support
         new and experimental type definitions.";
      reference
        "RFC 5701: IPv6 Address Specific BGP Extended Community
         Attribute, Section 3.";
    }

    /* BGP Extended Community Types. */
```

```
    typedef bgp-ext-community-type {
      type union {
        type string {
          pattern 'route\-target:(6[0-5][0-5][0-3][0-5]|'
                + '[1-5][0-9]{4}|[1-9][0-9]{1,4}|[0-9]):'
                + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
                + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[0-9])';
          /*
           * description
           *   "Type 0x00, Sub-Type 0x02: Route-Target
           *    route-target:(ASN):(local-admin)
           *    2 octets global administrator and 4 octets local
           *    administrator.";
           * reference
           *   "RFC 4360: BGP Extended Communities Attribute,
           *    Section 4.";
           */
        }

        type string {
          pattern 'route\-target:'
                + '(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|'
                + '25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|'
                + '2[0-4][0-9]|25[0-5]):'
                + '(6[0-5][0-5][0-3][0-5]|[1-5][0-9]{4}|'
                + '[1-9][0-9]{1,4}|[0-9])';
          /*
           * description
           *   "Type 0x01, Sub-Type 0x02: Route-Target
           *    route-target:(IPv4):(local-admin)
           *    4 octets IP address global administrator and 2 octets
           *    local administrator.";
           * reference
           *   "RFC 4360: BGP Extended Communities Attribute,
           *    Section 4.";
           */
        }

        type string {
          pattern 'route\-target:'
                + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
                + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[0-9])'
                + '(6[0-5][0-5][0-3][0-5]|[1-5][0-9]{4}|'
                + '[1-9][0-9]{1,4}|[0-9])';
          /*
           * description
           *   "Type 0x02, Sub-Type 0x02: Route-Target
           *    route-target:(ASN):(local-admin)
```

```
            *    4 octets global administrator and 2 octets local
            *    administrator.";
            * reference
            *   "RFC 5668: 4-Octet AS Specific BGP Extended Community,
            *    Section 4.";
            */
          }

          type string {
            pattern 'route\-origin:(6[0-5][0-5][0-3][0-5]|'
                  + '[1-5][0-9]{4}|[1-9][0-9]{1,4}|[0-9]):'
                  + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
                  + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[0-9])';
            /*
             * description
             *   "Type 0x01, Sub-Type 0x03: Route-Origin
             *    route-target:(ASN):(local-admin)
             *    2 octets global administrator and 4 octets local
             *    administrator.";
             * reference
             *   "RFC 4360: BGP Extended Communities Attribute,
             *    Section 5.";
             */
          }

          type string {
            pattern 'route\-origin:'
                  + '(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|'
                  + '25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|'
                  + '2[0-4][0-9]|25[0-5]):'
                  + '(6[0-5][0-5][0-3][0-5]|[1-5][0-9]{4}|'
                  + '[1-9][0-9]{1,4}|[0-9])';
            /*
             * description
             *   "Type 0x01, Sub-Type 0x03: Route-Target
             *    route-target:(IPv4):(local-admin)
             *    4 octets IP address global administrator and 2 octets
             *    local administrator.";
             * reference
             *   "RFC 4360: BGP Extended Communities Attribute,
             *    Section 5.";
             */
          }

          type string {
            pattern 'route\-origin:'
                  + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
                  + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[0-9])'
```

```
                + '(6[0-5][0-5][0-3][0-5]|[1-5][0-9]{4}|'
                + '[1-9][0-9]{1,4}|[0-9])';
            /*
             * description
             *   "Type 0x02, Sub-Type 0x03: Route-Origin
             *    route-origin:(ASN):(local-admin)
             *    4 octets global administrator and 2 octets local
             *    administrator.";
             * reference
             *   "RFC 5668: 4-Octet AS Specific BGP Extended Community,
             *    Section 4.";
             */
          }

          type string {
            // raw with 8 octets
            pattern 'raw:'
                 + '([0-9A-Fa-f][0-9A-Fa-f]:){7}'
                 + '[0-9A-Fa-f][0-9A-Fa-f]';
          }
        }
        description
          "Type definition for extended community attributes.
           It includes a way to specify a 'raw' string that
           is followed by 8 bytes of octet string to support
           new and experimental type definitions.";
        reference
          "RFC 4360: BGP Extended Communities Attribute.";
      }
    }
    <CODE ENDS>
```

7.5. BGP policy module

```
    <CODE BEGINS> file "ietf-bgp-policy@2023-07-05.yang"
    module ietf-bgp-policy {
      yang-version 1.1;
      namespace "urn:ietf:params:xml:ns:yang:ietf-bgp-policy";
      prefix bp;

      // import some basic types

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

```
      prefix rt-pol;
      reference
        "RFC 9067: A YANG Data Model for Routing Policy";
    }
    import iana-bgp-types {
      prefix bt;
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }
    import iana-bgp-community-types {
      prefix bct;
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    organization
      "IETF IDR Working Group";
    contact
      "WG Web:   <http://datatracker.ietf.org/wg/idr>
       WG List:  <idr@ietf.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com),
                Jeffrey Haas (jhaas at juniper.net).";

    description
      "This module contains data definitions for BGP routing policy.
       It augments the base routing-policy module with BGP-specific
       options for conditions and actions.

       Copyright (c) 2023 IETF Trust and the persons identified as
       authors of the code. All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Revised BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (https://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX
       (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
       for full legal notices.

       The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
       NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
       'MAY', and 'OPTIONAL' in this document are to be interpreted as
```

```
        described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
        they appear in all capitals, as shown here.";

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    /*
     * typedef statements
     */

    typedef bgp-set-community-option-type {
      type enumeration {
        enum add {
          description
            "Add the specified communities to the existing
             community attribute.";
        }
        enum remove {
          description
            "Remove the specified communities from the
             existing community attribute.";
        }
        enum replace {
          description
            "Replace the existing community attribute with
             the specified communities. If an empty set is
             specified, this removes the community attribute
             from the route.";
        }
      }
      description
        "Type definition for options when setting the community
         attribute in a policy action.";
    }

    typedef bgp-next-hop-type {
      type union {
        type inet:ip-address-no-zone;
        type enumeration {
          enum self {
            description
              "Special designation for local router's own
               address, i.e., next-hop-self.";
          }
```

```
        }
      }
      description
        "Type definition for specifying next-hop in policy actions.";
    }

    typedef bgp-set-med-type {
      type union {
        type string {
          pattern '[+-]([0-9]{1,8}|[0-3][0-9]{1,9}|4[0-1][0-9]{1,8}|'
                + '428[0-9]{1,7}|429[0-3][0-9]{1,6}|42948[0-9]{1,5}|'
                + '42949[0-5][0-9]{1,4}|429496[0-6][0-9]{1,3}|'
                + '4294971[0-9]{1,2}|42949728[0-9]|42949729[0-5])';
        }
        type enumeration {
          enum igp {
            description
              "Set the MED value to the IGP cost toward the
               next hop for the route.";
          }
          enum med-plus-igp {
            description
              "Before comparing MED values for path selection, adds to
               the MED the cost of the IGP route to the BGP next-hop
               destination.

               This option replaces the MED value for the router,
               but does not affect the IGP metric comparison. As a
               result, when multiple routes have the same value
               after the MED-plus-IPG comparison, and route selection
               continues, the IGP route metric is also compared, even
               though it was added to the MED value and compared
               earlier in the selection process.

               Useful when the downstream AS requires the complete
               cost of a certain route that is received across
               multiple ASs.";
          }
        }
        type uint32;
      }
      description
        "Type definition for specifying how the BGP MED can
         be set in BGP policy actions. The three choices are to set
         the MED directly, increment/decrement using +/- notation,
         and setting it to the IGP cost (predefined value).";
    }
```

```
    grouping equality-operator {
      description
        "Grouping used for common equality operations in BGP policy.";
      choice operation {
        case eq {
          leaf eq {
            type empty;
            description
              "Check to see if the value is equal.";
          }
        }

        case lt-or-eq {
          leaf lt-or-eq {
            type empty;
            description
              "Check to see if the value is less than or equal.";
          }
        }

        case gt-or-eq {
          leaf gt-or-eq {
            type empty;
            description
              "Check to see if the value is greater than or
               equal.";
          }
        }
        description
          "Choice of operations on the value";
      }
    }

    // Identities

    // augment statements

    augment "/rt-pol:routing-policy/rt-pol:defined-sets" {
      description
        "Adds BGP defined sets container to routing policy model.";
      container bgp-defined-sets {
        description
          "BGP-related set definitions for policy match conditions.";
        container as-path-sets {
          description
            "Enclosing container for list of define AS path sets.";
          list as-path-set {
            key "name";
```

```
        description
          "List of defined AS path sets.";
        leaf name {
          type string;
          description
            "Name of the AS path set -- this is used to reference
             the set in match conditions.";
        }
        leaf-list member {
          type string;
          description
            "AS path regular expression using BGP YANG AS_PATH
             regular expression syntax.  If any of the regular
             expressions in the lists are matched, the as-path-set
             is considered matched.";
          reference
            "RFC XXXX: YANG Model for Border Gateway Protocol
             (BGP-4), Appendix F.3.";
        }
      }
    }

    container community-sets {
      description
        "Enclosing container for list of defined BGP community
         sets.";
      list community-set {
        key "name";
        description
          "List of defined BGP community sets.";
        leaf name {
          type string;
          description
            "Name / label of the community set -- this is used to
             reference the set in match conditions.";
        }
        leaf-list member {
          type union {
            type bct:bgp-well-known-community-type;
            type bct:bgp-std-community-type;
            type bct:bgp-community-regexp-type;
          }
          description
            "Members of the community set";
        }
      }
    }
```

```
        container ext-community-sets {
          description
            "Enclosing container for list of extended BGP community
             sets";
          list ext-community-set {
            key "name";
            description
              "List of defined extended BGP community sets";
            leaf name {
              type string;
              description
                "Name / label of the extended community set -- this is
                 used to reference the set in match conditions";
            }
            leaf-list member {
              type union {
                type bct:bgp-ext-community-type;
                type bct:bgp-community-regexp-type;
              }
              description
                "Members of the extended community set.";
            }
          }
        }

        container ipv6-ext-community-sets {
          description
            "Enclosing container for list of extended IPv6 BGP
             community sets";
          list ipv6-ext-community-set {
            key "name";
            description
              "List of defined IPv6 extended BGP community sets";
            leaf name {
              type string;
              description
                "Name / label of the IPv6 extended community set --
                 this is used to reference the set in match
                 conditions";
            }
            leaf-list member {
              type union {
                type bct:bgp-ipv6-ext-community-type;
                type bct:bgp-community-regexp-type;
              }
              description
                "Members of the IPv6 extended community set.";
            }
```

```
          }
        }

        container large-community-sets {
          description
            "Enclosing container for list of large BGP community
             sets";
          list large-community-set {
            key "name";
            description
              "List of defined large BGP community sets";
            leaf name {
              type string;
              description
                "Name / label of the large community set -- this is
                 used to reference the set in match conditions";
            }
            leaf-list member {
              type union {
                type bct:bgp-large-community-type;
                type bct:bgp-community-regexp-type;
              }
              description
                "Members of the large community set.";
            }
          }
        }

        container next-hop-sets {
          description
            "Definition of a list of IPv4 or IPv6 next-hops which can
             be matched in a routing policy.";

          list next-hop-set {
            key "name";
            description
              "List of defined next-hop sets for use in policies.";

            leaf name {
              type string;
              description
                "Name of the next-hop set.";
            }
            leaf-list next-hop {
              type bgp-next-hop-type;
              description
                "List of IP addresses in the next-hop set.";
            }
```

```
          }
        }
      }
    }

    augment "/rt-pol:routing-policy/rt-pol:policy-definitions/" +
            "rt-pol:policy-definition/rt-pol:statements/" +
            "rt-pol:statement/rt-pol:conditions" {
      description
        "BGP policy conditions added to routing policy module.";

      container bgp-conditions {
        description
          "Top-level container for BGP specific policy conditions.";

        container local-pref {
          description
            "Value and comparison operations for conditions based on
             the value of the BGP LOCAL_PREF.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
             5.1.5.";

          leaf value {
            type uint32;
            description
              "BGP LOCAL_PREF value for comparison to the entry in the
               BGP route.";
          }

          uses equality-operator;
        }

        container med {
          description
            "Value and comparison operations for conditions based on
             the value of the BGP MULTI_EXIT_DISC (MED)";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
             5.1.4.";

          leaf value {
            type uint32;
            description
              "BGP MED value for comparison to the entry in the BGP
               route.";
          }
```

```
          uses equality-operator;
        }

        leaf origin-eq {
          type bt:bgp-origin-attr-type;
          description
            "Condition to check if the route origin is equal to the
             specified value.";
        }

        container match-afi-safi {
          description
            "Match an address family according to the logic defined in
             the match-set-options leaf.";
          leaf-list afi-safi-in {
            type identityref {
              base bt:afi-safi-type;
            }
            description
              "List of address families which the NLRI may be within.";
          }
          uses rt-pol:match-set-options-restricted-group;
        }

        container match-neighbor {
          description
            "Match a neighbor according to the logic defined in the
             match-set-options leaf.";

          leaf-list neighbor-eq {
            type inet:ip-address;
            description
              "List of neighbor addresses to check for in the ingress
               direction.";
          }
          uses rt-pol:match-set-options-restricted-group;
        }

        leaf route-type {
          type enumeration {
            enum internal {
              description
                "route type is internal.";
            }
            enum external {
              description
                "route type is external.";
            }
```

```
          }
          description
            "Condition to check the route type in the route update.";
        }

        container community-count {
          description
            "Value and comparison operations for conditions based on
             the number of communities in the route update.";

          leaf community-count {
            type uint32;
            description
              "Value for the number of communities in the route
               update.";
          }

          uses equality-operator;
        }

        container as-path-length {
          description
            "Value and comparison operations for conditions based on
             the length of the AS path in the route update.

             The as-path-length SHALL be calculated and SHALL follow
             RFC 4271 rules.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4).";

          leaf as-path-length {
            type uint32;
            description
              "Value of the AS path length in the route update.";
          }

          uses equality-operator;
        }

        container match-community-set {
          description
            "Top-level container for match conditions on communities.
             Match a referenced community-set according to the logic
             defined in the match-set-options leaf.";
          leaf community-set {
            type leafref {
              path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                 + "bgp-defined-sets/community-sets/"
```

```
                + "community-set/name";
            }
            description
              "References a defined community set.";
          }
          uses rt-pol:match-set-options-group;
        }

        container match-ext-community-set {
          description
            "Match a referenced extended community-set according to the
             logic defined in the match-set-options leaf.";
          leaf ext-community-set {
            type leafref {
              path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                 + "bgp-defined-sets/ext-community-sets/"
                 + "ext-community-set/name";
            }
            description
              "References a defined extended community set.";
          }
          leaf ext-community-match-kind {
            type enumeration {
              enum ext-community {
                description
                  "Perform the match against the ext-community RIB
                   attribute.";
              }
              enum ext-community-raw {
                description
                  "Perform the match against the ext-community-raw RIB
                   attribute.";
              }
            }
            default ext-community;
            description
              "Extended communities may be matched by the ext-community
               RIB attribute, or the ext-community-raw RIB attribute.
               This leaf selects which leaf to perform the match
               operation against.";
          }
          uses rt-pol:match-set-options-group;
        }

        container match-ipv6-ext-community-set {
          description
            "Match a referenced IPv6 extended community-set according
             to the logic defined in the match-set-options leaf.";
```

```
          leaf ipv6-ext-community-set {
            type leafref {
              path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                 + "bgp-defined-sets/ipv6-ext-community-sets/"
                 + "ipv6-ext-community-set/name";
            }
            description
              "References a defined IPv6 extended community set.";
          }
          leaf ipv6-ext-community-match-kind {
            type enumeration {
              enum ipv6-ext-community {
                description
                  "Perform the match against the IPv6 ext-community RIB
                   attribute.";
              }
              enum ipv6-ext-community-raw {
                description
                  "Perform the match against the IPv6 ext-community-raw
                   RIB attribute.";
              }
            }
            default ipv6-ext-community;
            description
              "IPv6 Extended communities may be matched by the
               ipv6-ext-community RIB attribute, or the
               ipv6-ext-community-raw RIB attribute.  This leaf selects
               which leaf to perform the match operation against.";
          }
          uses rt-pol:match-set-options-group;
        }

        container match-large-community-set {
          description
            "Match a referenced large community-set according to the
             logic defined in the match-set-options leaf.";
          leaf large-community-set {
            type leafref {
              path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                 + "bgp-defined-sets/large-community-sets/"
                 + "large-community-set/name";
            }
            description
              "References a defined large community set.";
          }
          uses rt-pol:match-set-options-group;
        }
```

```
        container match-as-path-set {
          description
            "Match a referenced as-path set according to the logic
             defined in the match-set-options leaf.";
          leaf as-path-set {
            type leafref {
              path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                 + "bgp-defined-sets/as-path-sets/"
                 + "as-path-set/name";
            }
            description
              "References a defined AS path set";
          }
          uses rt-pol:match-set-options-group;
        }

        container match-next-hop-set {
          description
            "Match a referenced next-hop set according to the logic
             defined in the match-set-options leaf.";
          leaf next-hop-set {
            type leafref {
              path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                 + "bgp-defined-sets/next-hop-sets/"
                 + "next-hop-set/name";
            }
            description
              "Reference a defined next-hop set.";
          }
          uses rt-pol:match-set-options-restricted-group;
        }
      }
    }

    augment "/rt-pol:routing-policy/rt-pol:policy-definitions/" +
            "rt-pol:policy-definition/rt-pol:statements/" +
            "rt-pol:statement/rt-pol:actions" {
      description
        "BGP policy actions added to routing policy module.";
      container bgp-actions {
        description
          "Top-level container for BGP-specific actions";
        leaf set-route-origin {
          type bt:bgp-origin-attr-type;
          description
            "Set the origin attribute to the specified value";
        }
        leaf set-local-pref {
```

```
          type uint32;
          description
            "Set the local pref attribute on the route.";
        }
        leaf set-next-hop {
          type bgp-next-hop-type;
          description
            "Set the next-hop attribute in the route.";
        }
        leaf set-med {
          type bgp-set-med-type;
          description
            "Set the med metric attribute in the route.";
        }

        container set-as-path-prepend {
          description
            "Action to prepend local AS number to the AS-path a
             specified number of times";

          leaf repeat-n {
            type uint8 {
              range "1..max";
            }
            description
              "Number of times to prepend the AS number to the AS
               path. The value should be between 1 and the maximum
               supported by the implementation.";
          }
          leaf-list asn {
            type inet:as-number;
            description
              "AS number to prepend to the AS path.";
          }
        }

        container set-community {
          description
            "Action to set the community attributes of the route, along
             with options to modify how the community is modified.
             Communities may be set using an inline list OR
             reference to an existing defined set (not both).";

          leaf options {
            type bgp-set-community-option-type;
            description
              "Options for modifying the community attribute with
               the specified values.  These options apply to both
```

```
            methods of setting the community attribute.";
        }

        choice method {
          description
            "Indicates the method used to specify the extended
             communities for the set-community action";
          case inline {
            leaf-list communities {
              type union {
                type bct:bgp-well-known-community-type;
                type bct:bgp-std-community-type;
              }
              description
                "Set the community values for the update inline with
                 a list.";
            }
          }

          case reference {
            leaf community-set-ref {
              type leafref {
                path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                   + "bgp-defined-sets/"
                   + "community-sets/community-set/name";
              }
              description
                "References a defined community set by name";
            }
          }
        }
      }

      container set-ext-community {
        description
          "Action to set the extended community attributes of the
           route, along with options to modify how the community is
           modified. Extended communities may be set using an inline
           list OR a reference to an existing defined set (but not
           both).";

        leaf options {
          type bgp-set-community-option-type;
          description
            "Options for modifying the community attribute with
             the specified values.  These options apply to both
             methods of setting the community attribute.";
        }
```

```
            choice method {
              description
                "Indicates the method used to specify the extended
                 communities for the set-ext-community action";
              case inline {
                leaf-list communities {
                  type bct:bgp-ext-community-type;
                  description
                    "Set the extended community values for the update
                     inline with a list.";
                }
              }
              case reference {
                leaf ext-community-set-ref {
                  type leafref {
                    path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                       + "bgp-defined-sets/ext-community-sets/"
                       + "ext-community-set/name";
                  }
                  description
                    "References a defined extended community set by
                     name.";
                }
              }
            }
          }

          container set-ipv6-ext-community {
            description
              "Action to set the IPv6 extended community attributes of
               the route, along with options to modify how the community
               is modified. IPv6 extended communities may be set using an
               inline list OR a reference to an existing defined set (but
               not both).";

            leaf options {
              type bgp-set-community-option-type;
              description
                "Options for modifying the community attribute with
                 the specified values.  These options apply to both
                 methods of setting the community attribute.";
            }

            choice method {
              description
                "Indicates the method used to specify the IPv6 extended
                 communities for the set-ipv6-ext-community action";
              case inline {
```

```
            leaf-list communities {
              type bct:bgp-ipv6-ext-community-type;
              description
                "Set the IPv6 extended community values for the
                 update inline with a list.";
            }
          }
          case reference {
            leaf ipv6-ext-community-set-ref {
              type leafref {
                path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                   + "bgp-defined-sets/ipv6-ext-community-sets/"
                   + "ipv6-ext-community-set/name";
              }
              description
                "References a defined IPv6 extended community set by
                 name.";
            }
          }
        }
      }

      container set-large-community {
        description
          "Action to set the large community attributes of the
           route, along with options to modify how the community is
           modified. Large communities may be set using an inline
           list OR a reference to an existing defined set (but not
           both).";

        leaf options {
          type bgp-set-community-option-type;
          description
            "Options for modifying the community attribute with
             the specified values. These options apply to both
             methods of setting the community attribute.";
        }

        choice method {
          description
            "Indicates the method used to specify the large
             communities for the set-large-community action";
          case inline {
            leaf-list communities {
              type bct:bgp-large-community-type;
              description
                "Set the large community values for the update
                 inline with a list.";
```

```
              }
            }
            case reference {
              leaf large-community-set-ref {
                type leafref {
                  path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                     + "bgp-defined-sets/large-community-sets/"
                     + "large-community-set/name";
                }
                description
                  "References a defined extended community set by
                   name.";
              }
            }
          }
        }
      }
    }
  }
  <CODE ENDS>
```

7.6. RIB modules and submodules

7.6.1. BGP RIB types module

```
  <CODE BEGINS> file "iana-bgp-rib-types@2023-07-05.yang"
  module iana-bgp-rib-types {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:iana-bgp-rib-types";
    prefix brt;

    organization
      "IANA";
    contact
      "Internet Assigned Numbers Authority

       Postal: ICANN
               12025 Waterfront Drive, Suite 300
               Los Angeles, CA 90094-2536
               United States of America
       Tel:    +1 310 301 5800
       <mailto:iana@iana.org>

       Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                Keyur Patel (keyur at arrcus.com),
                Susan Hares (shares at ndzh.com),
                Jeffrey Haas (jhaas at juniper.net).";
```

```
    description
      "Defines identity and type definitions associated with
       the BGP RIB modules.

       Copyright (c) 2023 IETF Trust and the persons identified as
       authors of the code. All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Revised BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (https://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX
       (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
       for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    identity ineligible-route-reason {
      description
        "Base identity for reason code for routes that are rejected as
         ineligible. Some derived entities are based on BMP v3.";
      reference
        "RFC 7854: BGP Monitoring Protocol.";
    }

    identity ineligible-cluster-loop {
      base ineligible-route-reason;
      description
        "Route was ineligible due to CLUSTER_LIST loop";
    }

    identity ineligible-as-loop {
      base ineligible-route-reason;
      description
```

```
        "Route was ineligible due to AS_PATH loop";
    }

    identity ineligible-originator {
      base ineligible-route-reason;
      description
        "Route was ineligible due to ORIGINATOR_ID. For example, update
         has local router as originator";
    }

    identity ineligible-confed {
      base ineligible-route-reason;
      description
        "Route was ineligible due to a loop in the AS_CONFED_SEQUENCE
         or AS_CONFED_SET attributes";
    }

    identity bgp-not-selected-bestpath {
      description
        "Base identity for indicating reason a route was was not
         selected by BGP route selection algorithm";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section 9.1.";
    }

    identity local-pref-lower {
      base bgp-not-selected-bestpath;
      description
        "Route has a lower localpref attribute than current best path";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 9.1.2.";
    }

    identity as-path-longer {
      base bgp-not-selected-bestpath;
      description
        "Route has a longer AS path attribute than current best path";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 9.1.2.2 (a).";
    }

    identity origin-type-higher {
      base bgp-not-selected-bestpath;
      description
        "Route has a higher origin type, i.e., IGP origin is preferred
         over EGP or incomplete";
```

```
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 9.1.2.2 (b).";
    }

    identity med-higher {
      base bgp-not-selected-bestpath;
      description
        "Route has a higher MED, or metric, attribute than the current
         best path";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 9.1.2.2 (c).";
    }

    identity prefer-external {
      base bgp-not-selected-bestpath;
      description
        "Route source is via IBGP, rather than EGP.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 9.1.2.2 (d).";
    }

    identity nexthop-cost-higher {
      base bgp-not-selected-bestpath;
      description
        "Route has a higher interior cost to the next hop.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 9.1.2.2 (e).";
    }

    identity higher-router-id {
      base bgp-not-selected-bestpath;
      description
        "Route was sent by a peer with a higher BGP Identifier value.";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
         Section 9.1.2.2 (f).";
    }

    identity higher-peer-address {
      base bgp-not-selected-bestpath;
      description
        "Route was sent by a peer with a higher IP address";
      reference
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
```

```
          Section 9.1.2.2 (g).";
      }

      identity bgp-not-selected-policy {
        description
          "Base identity for reason code for routes that are rejected
           due to policy";
      }

      identity rejected-import-policy {
        base bgp-not-selected-policy;
        description
          "Route was rejected after applying import policies.";
      }
    }
    <CODE ENDS>
```

7.6.2. BGP RIB submodule

```
    <CODE BEGINS> file "ietf-bgp-rib@2023-07-05.yang"
    submodule ietf-bgp-rib {
      yang-version 1.1;
      belongs-to ietf-bgp {
        prefix bgp;
      }

      /*
       * Import and Include
       */

      import iana-bgp-types {
        prefix bt;
        reference
          "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
      }
      import ietf-inet-types {
        prefix inet;
        reference
          "RFC 6991: Common YANG Types.";
      }
      import ietf-yang-types {
        prefix yang;
        reference
          "RFC 6991: Common YANG Types.";
      }
      include ietf-bgp-rib-tables;

      // groupings of attributes in three categories:
```

```
     //  - shared across multiple routes
     //  - common to Loc-RIB and Adj-RIB, but not shared across routes
     //  - specific to Loc-RIB or Adj-RIB
     // groupings of annotations for each route or table
     include ietf-bgp-rib-attributes;

     organization
       "IETF IDR Working Group";
     contact
       "WG Web:   <http://datatracker.ietf.org/wg/idr>
        WG List:  <idr@ietf.org>

        Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
                 Keyur Patel (keyur at arrcus.com),
                 Susan Hares (shares at ndzh.com),
                 Jeffrey Haas (jhaas at juniper dot net).";

     description
       "Defines a submodule for representing BGP Routing Information
        Base (RIB) contents.  The submodule supports 5 logical RIBs per
        address family:

        loc-rib: This is the main BGP routing table for the local
        routing instance, containing best-path selections for each
        prefix.  The key for the loc-rib will correspond either to a
        route in the adj-rib-in-post table, described below, or a route
        redistributed from another protocol.

        adj-rib-in-pre: This is a per-neighbor table containing the BGP
        routes received from the neighbor before any local input policy
        rules has been applied.  This can be considered the 'raw' routes
        from a given neighbor.

        adj-rib-in-post: This table is a per-neighbor table containing
        the routes received from the neighbor that are eligible for
        best-path selection after local input policy rules have been
        applied. The 'best-path' leaf is attached to the route selected
        by the BGP Decision Process (RFC 4271, Section 9.1).  Such
        routes may be present in the loc-rib table, described above,
        when local configuration determines that the BGP best-path will
        be used for that destination.

        adj-rib-in-post: This is a per-neighbor table containing the
        routes received from the neighbor that are eligible for
        best-path selection after local input policy rules have been
        applied.

        adj-rib-out-pre: This is a per-neighbor table containing routes
```

```
        eligible for sending (advertising) to the neighbor before output
        policy rules have been applied.

        adj-rib-out-post: This is a per-neighbor table containing routes
        eligible for sending (advertising) to the neighbor after output
        policy rules have been applied.

        Copyright (c) 2023 IETF Trust and the persons identified as
        authors of the code. All rights reserved.

        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject to
        the license terms contained in, the Revised BSD License set
        forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.

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

      revision 2023-07-05 {
        description
          "Initial Version";
        reference
          "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
      }

      grouping attr-set-attributes {
        description
          "A grouping for all BGP Path Attribute set parameters.";

        container attributes {
          description
            "A container for attribute set parameters.";

          leaf origin {
            type bt:bgp-origin-attr-type;
            description
              "BGP attribute defining the origin of the path
               information.";
            reference
```

```
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
             5.1.1.";
          }
          container as-path {
            description
              "Enclosing container for the list of AS path segments.

               In the Adj-RIB-In or Adj-RIB-Out, this list should show
               the received or sent AS_PATH, respectively.  For
               example, if the local router is not 4-byte capable, this
               value should consist of 2-octet ASNs or the AS_TRANS
               (AS 23456) values received or sent in BGP updates.

               In the Loc-RIB, this list should reflect the effective
               AS path for the route, e.g., a 4-octet value if the
               local router is 4-octet capable.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
               RFC 6793: BGP Support for Four-octet AS Number Space
               RFC 5065: Autonomous System Confederations for BGP.";
            list segment {
              config false;
              uses bgp-as-path-segment;
              description
                "List of AS PATH segments";
            }
          }
          leaf next-hop {
            type inet:ip-address;
            description
              "BGP next hop attribute defining the IP address of the
               router that should be used as the next hop to the
               destination.  Used when the BGP routes' nexthop for that
               AFI/SAFI can be represented as an IPv4/IPv6 address.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
               5.1.3.";
          }
          leaf link-local-next-hop {
            type inet:ipv6-address;
            description
              "When both a global and a link-local next-hop are sent
               when following RFC 2545 procedures, this leaf contains
               the link-local next-hop.";
            reference
              "RFC 2545: Use of BGP-4 Multiprotocol Extensions for IPv6
               Inter-Domain Routing.";
          }
```

```
        leaf med {
          type uint32;
          description
            "BGP multi-exit discriminator attribute used in the BGP
             route selection process.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
             5.1.4.";
        }
        leaf local-pref {
          type uint32;
          description
            "BGP local preference attribute sent to internal peers to
             indicate the degree of preference for externally learned
             routes.  The route with the highest local preference
             value is preferred.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
             5.1.5.";
        }
        container as4-path {
          description
            "This is the path encoded with 4-octet
             AS numbers in the optional transitive AS4_PATH attribute.
             This value is populated with the received or sent
             attribute in Adj-RIB-In or Adj-RIB-Out, respectively.
             It should not be populated in Loc-RIB since the Loc-RIB
             is expected to store the effective AS-Path in the
             as-path leaf regardless of being 4-octet or 2-octet.";
          reference
            "RFC 6793: BGP Support for Four-octet AS Number Space";
          list segment {
            config false;
            uses bgp-as-path-segment;
            description
              "List of AS PATH segments";
          }
        }
        container aggregator {
          config false;
          description
            "BGP attribute indicating the prefix has been
             aggregated by the specified AS and router.";
          reference
            "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
             5.1.7,
             RFC 6793: BGP Support for Four-octet AS Number Space.";
          leaf as {
```

```
            type inet:as-number;
            description
              "AS number of the autonomous system that performed the
               aggregation.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
               4.3, Path Attributes (g).";
          }
          leaf identifier {
            type yang:dotted-quad;
            description
              "BGP Identifier of the router that performed the
               aggregation.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
               4.3, Path Attributes (g).";
          }
        }
        container aggregator4 {
          config false;
          description
            "BGP attribute indicating the prefix has been
             aggregated by the specified AS and router.
             This value is populated with the received or sent
             attribute in Adj-RIB-In or Adj-RIB-Out, respectively.
             It should not be populated in Loc-RIB since the Loc-RIB
             is expected to store the effective AGGREGATOR in the
             aggregator/as leaf regardless of being 4-octet or
             2-octet.";
          reference
            "RFC 6793: BGP Support for Four-octet AS Number Space.";
          leaf as4 {
            type inet:as-number;
            description
              "AS number of the autonomous system that performed the
               aggregation (4-octet representation).  This value is
               populated if an upstream router is not 4-octet capable.
               Its semantics are similar to the AS4_PATH optional
               transitive attribute";
            reference
              "RFC 6793: BGP Support for Four-octet AS Number Space,
               Section 3,
               RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
               4.3, Path Attributes (g).";
          }
          leaf identifier {
            type yang:dotted-quad;
            description
```

```
            "BGP Identifier of the router that performed the
             aggregation.";
          reference
            "RFC 6793: BGP Support for Four-octet AS Number Space,
             Section 3,
             RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
             4.3, Path Attributes (g).";
        }
      }
      leaf atomic-aggregate {
        type boolean;
        description
          "BGP attribute indicating that the prefix is an atomic
           aggregate; i.e., the peer selected is a less specific
           route without selecting a more specific route that is
           subsumed by it.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4), Section
           5.1.6.";
      }
      leaf originator-id {
        type yang:dotted-quad;
        description
          "BGP attribute that provides the id as an IPv4 address
           of the originator of the announcement.";
        reference
          "RFC 4456: BGP Route Reflection: An Alternative to Full
           Mesh Internal BGP (IBGP).";
      }
      leaf-list cluster-list {
        type yang:dotted-quad;
        description
          "Represents the reflection path that the route has
           passed.";
        reference
          "RFC 4456: BGP Route Reflection: An Alternative to Full
           Mesh Internal BGP (IBGP).";
      }
      leaf aigp-metric {
        type uint64;
        description
          "BGP path attribute representing the accumulated IGP
           metric for the path";
        reference
          "RFC 7311: The Accumulated IGP Metric Attribute for BGP.";
      }
    }
  }
```

```
    grouping attr-sets {
      description
        "A grouping for all sets of path attributes.";

      container attr-sets {
        description
          "Enclosing container for the list of path attribute sets";

        list attr-set {
          key "index";
          description
            "List of path attributes that may be in use by multiple
             routes in the table";
          leaf index {
            type uint64;
            description
              "System generated index for each attribute set.  The
               index is used to reference an attribute set from a
               specific path.  Multiple paths may reference the same
               attribute set.";
          }
          uses attr-set-attributes;
        }
      }
    }

    grouping rib {
      description
        "Grouping for rib.";

      container rib {
        config false;
        uses attr-sets;
        container communities {
          description
            "Enclosing container for the list of community attribute
             sets.";
          list community {
            key "index";
            config false;
            description
              "List of path attributes that may be in use by multiple
               routes in the table.";
            leaf index {
              type uint64;
              description
                "System generated index for each attribute set.  The
                 index is used to reference an attribute set from a
```

```
                specific path.  Multiple paths may reference the same
                attribute set.";
            }
            uses bgp-community-attr-state;
          }
        }
        container ext-communities {
          description
            "Enclosing container for the list of extended community
             attribute sets.";
          list ext-community {
            key "index";
            config false;
            description
              "List of path attributes that may be in use by multiple
               routes in the table.";
            leaf index {
              type uint64;
              description
                "System generated index for each attribute set.  The
                 index is used to reference an attribute set from a
                 specific path.  Multiple paths may reference the same
                 attribute set.";
            }
            uses ext-community-attributes;
          }
        }
        container ipv6-ext-communities {
          description
            "Enclosing container for the list of IPv6 extended
             community attribute sets.";
          list ipv6-ext-community {
            key "index";
            config false;
            description
              "List of path attributes that may be in use by multiple
               routes in the table.";
            leaf index {
              type uint64;
              description
                "System generated index for each attribute set.  The
                 index is used to reference an attribute set from a
                 specific path.  Multiple paths may reference the same
                 attribute set.";
            }
            uses ipv6-ext-community-attributes;
          }
        }
```

```
        container large-communities {
          description
            "Enclosing container for the list of large community
             attribute sets.";
          list large-community {
            key "index";
            config false;
            description
              "List of path attributes that may be in use by multiple
               routes in the table.";
            leaf index {
              type uint64;
              description
                "System generated index for each attribute set. The
                 index is used to reference an attribute set from a
                 specific path.  Multiple paths may reference the same
                 attribute set.";
            }
            uses large-community-attributes;
          }
        }

        container afi-safis {
          config false;
          description
            "Enclosing container for address family list.";
          list afi-safi {
            key "name";
            description
              "List of afi-safi types.";
            leaf name {
              type identityref {
                base bt:afi-safi-type;
              }
              description
                "AFI,SAFI name.";
            }
            container ipv4-unicast {
              when "../name = 'bt:ipv4-unicast'" {
                description
                  "Include this container for IPv4 unicast RIB.";
              }
              description
                "Routing tables for IPv4 unicast -- active when the
                 afi-safi name is ipv4-unicast.";

              container loc-rib {
                config false;
```

```
        description
          "Container for the IPv4 Unicast BGP Loc-RIB data.";
        container routes {
          description
            "Enclosing container for list of routes in the
             routing table.";
          list route {
            key "prefix origin path-id";
            description
              "List of routes in the table, keyed by the route
               prefix, the route origin, and path-id. The route
               origin can be either the neighbor address from
               which the route was learned, or the source
               protocol that injected the route. The path-id
               distinguishes routes for the same prefix
               received from a neighbor (e.g., if add-paths is
               enabled).";
            leaf prefix {
              type inet:ipv4-prefix;
              description
                "The IPv4 prefix corresponding to the route.";
            }
            uses bgp-loc-rib-common-keys;
            uses bgp-loc-rib-common-attr-refs;
            uses bgp-common-route-annotations-state;
            uses bgp-unknown-attr-top;
            uses rib-ext-route-annotations;
          }
        }
      }

      container neighbors {
        config false;
        description
          "Enclosing container for neighbor list.";
        list neighbor {
          key "neighbor-address";
          description
            "List of neighbors (peers) of the local BGP
             speaker.";
          leaf neighbor-address {
            type inet:ip-address;
            description
              "IP address of the BGP neighbor or peer.";
          }
          container adj-rib-in-pre {
            description
              "Per-neighbor table containing the BGP routes
```

```
            received from the neighbor before any local
            input policy rules or filters have been applied.
            This can be considered the 'raw' routes from
            the neighbor.";
          uses ipv4-adj-rib-common;
          uses clear-routes {
            description
              "Clears the adj-rib-in state for the containing
               neighbor. Subsequently, implementations might
               issue a 'route refresh' if 'route refresh' has
               been negotiated, or reset the session. ";
          }
        }
        container adj-rib-in-post {
          description
            "Per-neighbor table containing the paths received
             from the neighbor that are eligible for
             best-path selection after local input policy
             rules have been applied.";
          uses ipv4-adj-rib-in-post;
          uses clear-routes {
            description
              "Clears the adj-rib-in state for the containing
               neighbor. Subsequently, implementations might
               issue a 'route refresh' if 'route refresh' has
               been negotiated, or reset the session. ";
          }
        }
        container adj-rib-out-pre {
          description
            "Per-neighbor table containing paths eligible for
             sending (advertising) to the neighbor before
             output policy rules have been applied.";
          uses ipv4-adj-rib-common;
          uses clear-routes {
            description
              "Clears the adj-rib-out state for the
               containing neighbor. Subsequently, neighbors
               will announce BGP updates to resynchronize
               these routes.";
          }
        }
        container adj-rib-out-post {
          description
            "Per-neighbor table containing paths eligible for
             sending (advertising) to the neighbor after
             output policy rules have been applied.";
          uses ipv4-adj-rib-common;
```

```
                uses clear-routes {
                  description
                    "Clears the adj-rib-out state for the
                     containing neighbor. Subsequently, neighbors
                     will announce BGP updates to resynchronize
                     these routes.";
                }
              }
            }
          }
        }

        container ipv6-unicast {
          when "../name = 'bt:ipv6-unicast'" {
            description
              "Include this container for IPv6 unicast RIB.";
          }
          description
            "Routing tables for IPv6 unicast -- active when the
             afi-safi name is ipv6-unicast.";

          container loc-rib {
            config false;
            description
              "Container for the IPv6 BGP Loc-RIB data.";
            container routes {
              description
                "Enclosing container for list of routes in the
                 routing table.";
              list route {
                key "prefix origin path-id";
                description
                  "List of routes in the table, keyed by the route
                   prefix, the route origin, and path-id. The route
                   origin can be either the neighbor address from
                   which the route was learned, or the source
                   protocol that injected the route. The path-id
                   distinguishes routes for the same prefix
                   received from a neighbor (e.g., if add-paths is
                   enabled).";
                leaf prefix {
                  type inet:ipv6-prefix;
                  description
                    "The IPv6 prefix corresponding to the route.";
                }
                uses bgp-loc-rib-common-keys;
                uses bgp-loc-rib-common-attr-refs;
                uses bgp-common-route-annotations-state;
```

```
        uses bgp-unknown-attr-top;
        uses rib-ext-route-annotations;
      }
    }
  }

  container neighbors {
    config false;
    description
      "Enclosing container for neighbor list.";
    list neighbor {
      key "neighbor-address";
      description
        "List of neighbors (peers) of the local BGP
         speaker.";
      leaf neighbor-address {
        type inet:ip-address;
        description
          "IP address of the BGP neighbor or peer.";
      }
      container adj-rib-in-pre {
        description
          "Per-neighbor table containing the BGP routes
           received from the neighbor before any local
           input policy rules or filters have been applied.
           This can be considered the 'raw' routes from
           the neighbor.";
        uses ipv6-adj-rib-common;
        uses clear-routes {
          description
            "Clears the adj-rib-in state for the containing
             neighbor. Subsequently, implementations might
             issue a 'route refresh' if 'route refresh' has
             been negotiated, or reset the session. ";
        }
      }
      container adj-rib-in-post {
        description
          "Per-neighbor table containing the paths received
           from the neighbor that are eligible for
           best-path selection after local input policy
           rules have been applied.";
        uses ipv6-adj-rib-in-post;
        uses clear-routes {
          description
            "Clears the adj-rib-in state for the containing
             neighbor. Subsequently, implementations might
             issue a 'route refresh' if 'route refresh' has
```

```
                        been negotiated, or reset the session. ";
                    }
                  }
                  container adj-rib-out-pre {
                    description
                      "Per-neighbor table containing paths eligible for
                       sending (advertising) to the neighbor before
                       output policy rules have been applied.";
                    uses ipv6-adj-rib-common;
                    uses clear-routes {
                      description
                        "Clears the adj-rib-out state for the
                         containing neighbor. Subsequently, neighbors
                         will announce BGP updates to resynchronize
                         these routes.";
                    }
                  }
                  container adj-rib-out-post {
                    description
                      "Per-neighbor table containing paths eligible for
                       sending (advertising) to the neighbor after
                       output policy rules have been applied.";
                    uses ipv6-adj-rib-common;
                    uses clear-routes {
                      description
                        "Clears the adj-rib-out state for the
                         containing neighbor. Subsequently, neighbors
                         will announce BGP updates to resynchronize
                         these routes.";
                    }
                  }
                }
              }
            }
          }
          description
            "Top level container for BGP RIB.";
        }
      }
    }
    <CODE ENDS>
```

7.6.3. BGP RIB attributes submodule

```
<CODE BEGINS> file "ietf-bgp-rib-attributes@2023-07-05.yang"
submodule ietf-bgp-rib-attributes {
  yang-version 1.1;
  belongs-to ietf-bgp {
    prefix bgp;
  }

  // import some basic types

  import iana-bgp-types {
    prefix bt;
    reference
      "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
  }
  import iana-bgp-community-types {
    prefix bct;
    reference
      "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
  }
  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types.";
  }

  // meta

  organization
    "IETF IDR Working Group";
  contact
    "WG Web:   <http://datatracker.ietf.org/wg/idr>
     WG List:  <idr@ietf.org>

     Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
              Keyur Patel (keyur at arrcus.com),
              Susan Hares (shares at ndzh.com),
              Jeffrey Haas (jhaas at juniper.net).";

  description
    "This submodule contains common data definitions for BGP
     attributes for use in BGP RIB tables.

     Copyright (c) 2023 IETF Trust and the persons identified as
     authors of the code. All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject to
     the license terms contained in, the Revised BSD License set
```

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

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    grouping bgp-as-path-segment {
      description
        "Data for representing BGP AS-PATH segment";

      leaf type {
        type identityref {
          base bt:as-path-segment-type;
        }
        description
          "The type of AS-PATH segment";
      }
      leaf-list member {
        type inet:as-number;
        description
          "List of the AS numbers in the AS-PATH segment";
      }
    }

    grouping bgp-unknown-attr-top {
      description
        "Unknown path attributes that are not expected to be shared
         across route entries, common to LOC-RIB and Adj-RIB";
      container unknown-attributes {
        description
          "Unknown path attributes that were received in the UPDATE
           message which contained the prefix.";
```

```
        list unknown-attribute {
          key "attr-type";
          description
            "This list contains received attributes that are
             unrecognized or unsupported by the local router. The list
             may be empty.";

          leaf optional {
            type boolean;
            description
              "Defines whether the attribute is optional (if
               set to true) or well-known (if set to false).
               Set in the high-order bit of the BGP attribute
               flags octet.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4).";
          }

          leaf transitive {
            type boolean;
            description
              "Defines whether an optional attribute is transitive
               (if set to true) or non-transitive (if set to false).
               For well-known attributes, the transitive flag must be
               set to true. Set in the second high-order bit of the BGP
               attribute flags octet.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4).";
          }

          leaf partial {
            type boolean;
            description
              "Defines whether the information contained in the
               optional transitive attribute is partial (if set to
               true) or complete (if set to false). For well-known
               attributes and for optional non-transitive attributes,
               the partial flag must be set to false. Set in the third
               high-order bit of the BGP attribute flags octet.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4).";
          }

          leaf extended {
            type boolean;
            description
              "Defines whether the attribute length is one octet
               (if set to false) or two octets (if set to true). Set in
```

```
              the fourth high-order bit of the BGP attribute flags
              octet.";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4).";
          }

          leaf attr-type {
            type uint8;
            description
              "1-octet value encoding the attribute type code";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4).";
          }

          leaf attr-len {
            type uint16;
            description
              "One or two octet attribute length field indicating the
               length of the attribute data in octets.  If the Extended
               Length attribute flag is set, the length field is 2
               octets, otherwise it is 1 octet";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4).";
          }

          leaf attr-value {
            type binary {
              length "0..65535";
            }
            description
              "Raw attribute value, not including the attribute
               flags, type, or length.  The maximum length
               of the attribute value data is 2^16-1 per the max value
               of the attr-len field (2 octets).";
            reference
              "RFC 4271: A Border Gateway Protocol 4 (BGP-4).";
          }
        }
      }
    }

    grouping bgp-adj-rib-attr-state {
      description
        "Path attributes that are not expected to be shared across
         route entries, specific to Adj-RIB";
      leaf path-id {
        type uint32;
        description
```

```
          "When the BGP speaker supports advertisement of multiple
           paths for a prefix, the path identifier is used to
           uniquely identify a route based on the combination of the
           prefix and path id.  In the Adj-RIB-In, the path-id value is
           the value received in the update message.   In the Loc-RIB,
           if used, it should represent a locally generated path-id
           value for the corresponding route.  In Adj-RIB-Out, it
           should be the value sent to a neighbor when add-paths is
           used, i.e., the capability has been negotiated.";
        reference
          "RFC 7911: Advertisement of Multiple Paths in BGP.";
      }
    }

    grouping bgp-community-attr-state {
      description
        "Common definition of BGP community attributes";
      leaf-list community {
        type union {
          type bct:bgp-well-known-community-type;
          type bct:bgp-std-community-type;
        }
        description
          "List of standard or well-known BGP community
           attributes.";
      }
    }

    grouping ext-community-attributes {
      description
        "A grouping for all extended community parameters.";

      leaf-list ext-community {
        type bct:bgp-ext-community-type;
        description
          "List of BGP extended community attributes. The received
           extended community may be an explicitly modeled
           type or unknown, represented by an 8-octet value
           formatted according to RFC 4360.";
        reference
          "RFC 4360: BGP Extended Communities Attribute.";
      }

      leaf-list ext-community-raw {
        type string {
          // raw with 8 octets
          pattern 'raw:([0-9A-F][0-9A-F]:){7}[0-9A-F][0-9A-F]';
        }
```

```
        description
          "ext-community type in raw format only.";
      }
    }

    grouping ipv6-ext-community-attributes {
      description
        "A grouping for all IPv6 extended community parameters.";

      leaf-list ipv6-ext-community {
        type bct:bgp-ipv6-ext-community-type;
        description
          "List of BGP IPv6 extended community attributes. The received
           IPv6 extended community may be an explicitly modeled
           type or unknown, represented by an 20-octet value
           formatted according to RFC 5701.";
        reference
          "RFC 5701: IPv6 Address Specific BGP Extended Community
           Attribute, Section 3.";
      }

      leaf-list ipv6-ext-community-raw {
        type string {
          // raw with 20 octets
          pattern 'ipv6-raw:'
                + '([0-9A-Fa-f][0-9A-Fa-f]:){19}'
                + '[0-9A-Fa-f][0-9A-Fa-f]';
        }
        description
          "IPv6 ext-community type in raw format only.";
      }
    }

    grouping large-community-attributes {
      description
        "A grouping for all large community parameters.";

      leaf-list large-community {
        type bct:bgp-large-community-type;
        description
          "List of BGP large community attributes.";
        reference
          "RFC 8092: BGP Large Communities Attribute.";
      }
    }
  }
  <CODE ENDS>
```

7.6.4. BGP RIB tables submodule

```
<CODE BEGINS> file "ietf-bgp-rib-tables@2023-07-05.yang"
submodule ietf-bgp-rib-tables {
  yang-version 1.1;
  belongs-to ietf-bgp {
    prefix bgp;
  }

  // import some basic types

  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types.";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types.";
  }
  import ietf-routing {
    prefix rt;
    reference
      "RFC 8022: A YANG Data Model for Routing Management.";
  }
  import iana-bgp-types {
    prefix bt;
    reference
      "RFC XXXX: YANG module for Border Gateway Protocol (BGP-4).";
  }
  import iana-bgp-rib-types {
    prefix brt;
    reference
      "RFC XXXX: YANG module for Border Gateway Protocol (BGP-4).";
  }
  include ietf-bgp-rib-attributes;

  organization
    "IETF IDR Working Group";
  contact
    "WG Web:   <http://datatracker.ietf.org/wg/idr>
     WG List:  <idr@ietf.org>

     Authors: Mahesh Jethanandani (mjethanandani at gmail.com),
              Keyur Patel (keyur at arrcus.com),
              Susan Hares (shares at ndzh.com,
              Jeffrey Haas (jhaas at juniper.net).";
```

```
    description
      "This submodule contains structural data definitions for
       BGP routing tables.

       Copyright (c) 2023 IETF Trust and the persons identified as
       authors of the code. All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Revised BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (https://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX
       (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
       for full legal notices.

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

    revision 2023-07-05 {
      description
        "Initial Version";
      reference
        "RFC XXXX: YANG Model for Border Gateway Protocol (BGP-4).";
    }

    grouping bgp-common-route-annotations-state {
      description
        "Data definitions for flags and other information attached
         to routes in both LOC-RIB and Adj-RIB";
      leaf last-modified {
        type yang:timeticks;
        description
          "Timestamp when this path was last modified.

           The value is the timestamp in seconds relative to
           the Unix Epoch (Jan 1, 1970 00:00:00 UTC).";
      }
      leaf eligible-route {
        type boolean;
        description
          "Indicates that the route is eligible for selection for the
           best route in the Loc-Rib in BGP's Decision Process.";
```

```
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
           Section 9.1.";
      }
      leaf ineligible-reason {
        type identityref {
          base brt:ineligible-route-reason;
        }
        description
          "If the route is ineligible for selection for the best route
           in the Loc-Rib in BGP's Decision process, this indicates the
           reason.";
        reference
          "RFC 4271: A Border Gateway Protocol 4 (BGP-4),
           Section 9.1.";
      }
    }

    grouping bgp-adj-rib-in-post-route-annotations-state {
      description
        "Data definitions for information attached to routes in the
         Adj-RIB-In post-policy table";
      leaf best-path {
        type boolean;
        default "false";
        description
          "Current path was selected as the best path. Best path
           should indicate that the route is present in BGP Loc-RIB.";
      }
    }

    grouping rib-ext-route-annotations {
      description
        "Extended annotations for routes in the routing tables";
      leaf reject-reason {
        type union {
          type identityref {
            base brt:bgp-not-selected-bestpath;
          }
          type identityref {
            base brt:bgp-not-selected-policy;
          }
        }
        description
          "Indicates the reason the route is not used, either due to
           policy filtering or bestpath selection";
      }
    }
```

```
    grouping bgp-adj-rib-common-attr-refs {
      description
        "Definitions of common references to attribute sets for
         multiple AFI-SAFIs for Adj-RIB tables.";
      leaf attr-index {
        type leafref {
          path "../../../../../../../../../attr-sets/"
             + "attr-set/index";
        }
        description
          "Reference to the common attribute group for the
           route.";
      }
      leaf community-index {
        type leafref {
          path "../../../../../../../../../communities/community/"
             + "index";
        }
        description
          "Reference to the community attribute for the route.";
      }
      leaf ext-community-index {
        type leafref {
          path "../../../../../../../../../ext-communities/"
             + "ext-community/index";
        }
        description
          "Reference to the extended community attribute for the
           route.";
      }
      leaf large-community-index {
        type leafref {
          path "../../../../../../../../../large-communities/"
             + "large-community/index";
        }
        description
          "Reference to the large community attribute for the
           route.";
      }
    }

    grouping bgp-loc-rib-common-attr-refs {
      description
        "Definitions of common references to attribute sets for
         multiple AFI-SAFIs for Loc-RIB tables.";
      leaf attr-index {
        type leafref {
          path "../../../../../../../attr-sets/attr-set/"
```

```
            + "index";
        }
        description
          "Reference to the common attribute group for the
           route.";
      }
      leaf community-index {
        type leafref {
          path "../../../../../../../communities/community/"
             + "index";
        }
        description
          "Reference to the community attribute for the route.";
      }
      leaf ext-community-index {
        type leafref {
          path "../../../../../../../ext-communities/"
             + "ext-community/index";
        }
        description
          "Reference to the extended community attribute for the
           route.";
      }
      leaf large-community-index {
        type leafref {
          path "../../../../../../../large-communities/"
             + "large-community/index";
        }
        description
          "Reference to the large community attribute for the
           route.";
      }
    }

    grouping bgp-loc-rib-common-keys {
      description
        "Common references used in keys for IPv4 and IPv6
         Loc-RIB entries.";
      leaf origin {
        type union {
          type inet:ip-address;
          type identityref {
            base rt:routing-protocol;
          }
        }
        description
          "Indicates the origin of the route.  If the route is learned
           from a neighbor, this value is the neighbor address.  If
```

```
            the route was injected or redistributed from another
            protocol, the origin indicates the source protocol for the
            route.";
        }
        leaf path-id {
          type uint32;
          description
            "If the route is learned from a neighbor, the path-id
             corresponds to the path-id for the route in the
             corresponding adj-rib-in-post table.  If the route is
             injected from another protocol, or the neighbor does not
             support BGP add-paths, the path-id should be set
             to zero, also the default value.

             However, YANG does not allow default values to be set
             for parameters that form the key, so a default value
             cannot be set here.";
        }
      }

      grouping clear-routes {
        description
          "Action to clear BGP routes.";
        container clear-routes {
          if-feature "bt:clear-routes";
          action clear {
            input {
              leaf clear-at {
                type yang:date-and-time;
                description
                  "The time, in the future when the clear operation will
                   be initiated.";
              }
            }
            output {
              leaf clear-finished-at {
                type yang:date-and-time;
                description
                  "The time when the clear operation finished.";
              }
            }
          }
          description
            "Action commands to clear routes governed by a if-feature.";
        }
      }

      grouping ipv4-adj-rib-common {
```

```
      description
        "Common structural grouping for each IPv4 Adj-RIB table.";
      container routes {
        config false;
        description
          "Enclosing container for list of routes in the routing
           table.";
        list route {
          key "prefix path-id";
          description
            "List of routes in the table, keyed by a combination of
             the route prefix and path-id to distinguish multiple
             routes received from a neighbor for the same prefix;
             e.g., when BGP add-paths is enabled.";
          leaf prefix {
            type inet:ipv4-prefix;
            description
              "Prefix for the route.";
          }
          uses bgp-adj-rib-attr-state;
          uses bgp-adj-rib-common-attr-refs;
          uses bgp-common-route-annotations-state;
          uses bgp-unknown-attr-top;
          uses rib-ext-route-annotations;
        }
      }
    }

    grouping ipv4-adj-rib-in-post {
      description
        "Common structural grouping for the IPv4 Adj-RIB-In
         post-policy table.";
      container routes {
        config false;
        description
          "Enclosing container for list of routes in the routing
           table.";
        list route {
          key "prefix path-id";
          description
            "List of routes in the table, keyed by a combination of
             the route prefix and path-id to distinguish multiple
             routes received from a neighbor for the same prefix;
             e.g., when BGP add-paths is enabled.";
          leaf prefix {
            type inet:ipv4-prefix;
            description
              "Prefix for the route.";
```

```
          }
          uses bgp-adj-rib-attr-state;
          uses bgp-adj-rib-common-attr-refs;
          uses bgp-common-route-annotations-state;
          uses bgp-adj-rib-in-post-route-annotations-state;
          uses bgp-unknown-attr-top;
          uses rib-ext-route-annotations;
        }
      }
    }

    grouping ipv6-adj-rib-common {
      description
        "Common structural grouping for each IPv6 Adj-RIB table.";
      container routes {
        config false;
        description
          "Enclosing container for list of routes in the routing
           table.";
        list route {
          key "prefix path-id";
          description
            "List of routes in the table.";
          leaf prefix {
            type inet:ipv6-prefix;
            description
              "Prefix for the route.";
          }
          uses bgp-adj-rib-attr-state;
          uses bgp-adj-rib-common-attr-refs;
          uses bgp-common-route-annotations-state;
          uses bgp-unknown-attr-top;
          uses rib-ext-route-annotations;
        }
      }
    }

    grouping ipv6-adj-rib-in-post {
      description
        "Common structural grouping for the IPv6 Adj-RIB-In
         post-policy table.";
      container routes {
        config false;
        description
          "Enclosing container for list of routes in the routing
           table.";
        list route {
          key "prefix path-id";
```

```
          description
            "List of routes in the table.";
          leaf prefix {
            type inet:ipv6-prefix;
            description
              "Prefix for the route.";
          }
          uses bgp-adj-rib-attr-state;
          uses bgp-adj-rib-common-attr-refs;
          uses bgp-common-route-annotations-state;
          uses bgp-adj-rib-in-post-route-annotations-state;
          uses bgp-unknown-attr-top;
          uses rib-ext-route-annotations;
        }
      }
    }
  }
  <CODE ENDS>
```

8. Contributors

Previous versions of this document saw contributions from Anees Shaikh, Rob Shakir, Kevin D'Souza, Alexander Clemm, Aleksandr Zhadkin, and Xufeng Liu.

9. Acknowledgements

The authors are grateful for valuable contributions to this document and the associated models from: Ebben Aires, Pavan Beeram, Chris Chase, Ed Crabbe, Luyuan Fang, Bill Fenner, Akshay Gattani, Josh George, Vijay Gill, Matt John, Dhanendra Jain, Acee Lindem, Ina Minei, Carl Moberg, Ashok Narayanan, Einar Nilsen-Nygaard, Adam Simpson, Puneet Sood, Jason Sterne, Jeff Tantsura, Jim Uttaro, and Gunter Vandevelde.

Credit is also due to authors of the OpenConfig, whose model was relied upon to come up with this model.

Special thanks to Robert Wilton who helped convert the YANG models to a NMDA compatible model, to Yingzhen Qu who contributed to the policy part of the module, and to Derek Yeung in helping out with Extended Communities support in the model.

10. References

10.1. Normative references

[RFC1997] Chandra, R., Traina, P., and T. Li, "BGP Communities Attribute", RFC 1997, DOI 10.17487/RFC1997, August 1996, <https://www.rfc-editor.org/info/rfc1997>.

[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>.

[RFC2439] Villamizar, C., Chandra, R., and R. Govindan, "BGP Route Flap Damping", RFC 2439, DOI 10.17487/RFC2439, November 1998, <https://www.rfc-editor.org/info/rfc2439>.

[RFC2918] Chen, E., "Route Refresh Capability for BGP-4", RFC 2918, DOI 10.17487/RFC2918, September 2000, <https://www.rfc-editor.org/info/rfc2918>.

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

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, DOI 10.17487/RFC4360, February 2006, <https://www.rfc-editor.org/info/rfc4360>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <https://www.rfc-editor.org/info/rfc4456>.

[RFC4659] De Clercq, J., Ooms, D., Carugi, M., and F. Le Faucheur, "BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN", RFC 4659, DOI 10.17487/RFC4659, September 2006, <https://www.rfc-editor.org/info/rfc4659>.

[RFC4724] Sangli, S., Chen, E., Fernando, R., Scudder, J., and Y. Rekhter, "Graceful Restart Mechanism for BGP", RFC 4724, DOI 10.17487/RFC4724, January 2007, <https://www.rfc-editor.org/info/rfc4724>.

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <https://www.rfc-editor.org/info/rfc4760>.

[RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <https://www.rfc-editor.org/info/rfc4761>.

[RFC5065] Traina, P., McPherson, D., and J. Scudder, "Autonomous System Confederations for BGP", RFC 5065, DOI 10.17487/RFC5065, August 2007, <https://www.rfc-editor.org/info/rfc5065>.

[RFC5701] Rekhter, Y., "IPv6 Address Specific BGP Extended Community Attribute", RFC 5701, DOI 10.17487/RFC5701, November 2009, <https://www.rfc-editor.org/info/rfc5701>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

[RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, DOI 10.17487/RFC5881, June 2010, <https://www.rfc-editor.org/info/rfc5881>.

[RFC5883] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for Multihop Paths", RFC 5883, DOI 10.17487/RFC5883, June 2010, <https://www.rfc-editor.org/info/rfc5883>.

[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>.

[RFC6514] Aggarwal, R., Rosen, E., Morin, T., and Y. Rekhter, "BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs", RFC 6514, DOI 10.17487/RFC6514, February 2012, <https://www.rfc-editor.org/info/rfc6514>.

[RFC6793] Vohra, Q. and E. Chen, "BGP Support for Four-Octet Autonomous System (AS) Number Space", RFC 6793, DOI 10.17487/RFC6793, December 2012, <https://www.rfc-editor.org/info/rfc6793>.

[RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", RFC 6811, DOI 10.17487/RFC6811, January 2013, <https://www.rfc-editor.org/info/rfc6811>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7607] Kumari, W., Bush, R., Schiller, H., and K. Patel, "Codification of AS 0 Processing", RFC 7607, DOI 10.17487/RFC7607, August 2015, <https://www.rfc-editor.org/info/rfc7607>.

[RFC7911] Walton, D., Retana, A., Chen, E., and J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, DOI 10.17487/RFC7911, July 2016, <https://www.rfc-editor.org/info/rfc7911>.

[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>.

[RFC8092] Heitz, J., Ed., Snijders, J., Ed., Patel, K., Bagdonas, I., and N. Hilliard, "BGP Large Communities Attribute", RFC 8092, DOI 10.17487/RFC8092, February 2017, <https://www.rfc-editor.org/info/rfc8092>.

[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>.

[RFC8177] Lindem, A., Ed., Qu, Y., Yeung, D., Chen, I., and J. Zhang, "YANG Data Model for Key Chains", RFC 8177, DOI 10.17487/RFC8177, June 2017, <https://www.rfc-editor.org/info/rfc8177>.

[RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017, <https://www.rfc-editor.org/info/rfc8277>.

[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>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

[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>.

[RFC8528] Bjorklund, M. and L. Lhotka, "YANG Schema Mount", RFC 8528, DOI 10.17487/RFC8528, March 2019, <https://www.rfc-editor.org/info/rfc8528>.

[RFC8529] Berger, L., Hopps, C., Lindem, A., Bogdanovic, D., and X. Liu, "YANG Data Model for Network Instances", RFC 8529, DOI 10.17487/RFC8529, March 2019, <https://www.rfc-editor.org/info/rfc8529>.

[RFC9067] Qu, Y., Tantsura, J., Lindem, A., and X. Liu, "A YANG Data Model for Routing Policy", RFC 9067, DOI 10.17487/RFC9067, October 2021, <https://www.rfc-editor.org/info/rfc9067>.

[RFC9293] Eddy, W., Ed., "Transmission Control Protocol (TCP)", STD 7, RFC 9293, DOI 10.17487/RFC9293, August 2022, <https://www.rfc-editor.org/info/rfc9293>.

[RFC9314] Jethanandani, M., Ed., Rahman, R., Ed., Zheng, L., Ed., Pallagatti, S., and G. Mirsky, "YANG Data Model for Bidirectional Forwarding Detection (BFD)", RFC 9314, DOI 10.17487/RFC9314, September 2022, <https://www.rfc-editor.org/info/rfc9314>.

[I-D.ietf-tcpm-yang-tcp]
Scharf, M., Jethanandani, M., and V. Murgai, "A YANG Model for Transmission Control Protocol (TCP) Configuration and State", Work in Progress, Internet-Draft, draft-ietf-tcpm-yang-tcp-09, 11 September 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-tcpm-yang-tcp-09>.

10.2. Informative references

[RFC2385] Heffernan, A., "Protection of BGP Sessions via the TCP MD5 Signature Option", RFC 2385, DOI 10.17487/RFC2385, August 1998, <https://www.rfc-editor.org/info/rfc2385>.

[RFC2622] Alaettinoglu, C., Villamizar, C., Gerich, E., Kessens, D., Meyer, D., Bates, T., Karrenberg, D., and M. Terpstra, "Routing Policy Specification Language (RPSL)", RFC 2622, DOI 10.17487/RFC2622, June 1999, <https://www.rfc-editor.org/info/rfc2622>.

[RFC3765] Huston, G., "NOPEER Community for Border Gateway Protocol (BGP) Route Scope Control", RFC 3765, DOI 10.17487/RFC3765, April 2004, <https://www.rfc-editor.org/info/rfc3765>.

[RFC4451] McPherson, D. and V. Gill, "BGP MULTI_EXIT_DISC (MED) Considerations", RFC 4451, DOI 10.17487/RFC4451, March 2006, <https://www.rfc-editor.org/info/rfc4451>.

[RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <https://www.rfc-editor.org/info/rfc5082>.

[RFC5398] Huston, G., "Autonomous System (AS) Number Reservation for Documentation Use", RFC 5398, DOI 10.17487/RFC5398, December 2008, <https://www.rfc-editor.org/info/rfc5398>.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <https://www.rfc-editor.org/info/rfc5925>.

[RFC6151] Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", RFC 6151, DOI 10.17487/RFC6151, March 2011, <https://www.rfc-editor.org/info/rfc6151>.

[RFC7454] Durand, J., Pepelnjak, I., and G. Doering, "BGP Operations and Security", BCP 194, RFC 7454, DOI 10.17487/RFC7454, February 2015, <https://www.rfc-editor.org/info/rfc7454>.

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

[I-D.ietf-idr-deprecate-as-set-confed-set] Kumari, W. A., Sriram, K., Hannachi, L., and J. Haas, "Deprecation of AS_SET and AS_CONFED_SET in BGP", Work in Progress, Internet-Draft, draft-ietf-idr-deprecate-as-set-confed-set-10, 16 January 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-deprecate-as-set-confed-set-10>.

Appendix A. Examples

This section tries to show some examples in how the model can be used.

A.1. Creating BGP Instance

This example shows how to enable BGP for a IPv4 unicast address family.

```
   [note: '\' line wrapping for formatting only]

   <?xml version="1.0" encoding="UTF-8"?>
   <routing
       xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
     <control-plane-protocols>
       <control-plane-protocol>
         <type
             xmlns:bgp="urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</\
   type>
         <name>BGP</name>
         <bgp
             xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
           <global>
             <as>64496</as>
             <afi-safis>
               <afi-safi>
                 <name
                     xmlns:bt=
                     "urn:ietf:params:xml:ns:yang:iana-bgp-types">bt:ip\
   v4-unicast</name>
               </afi-safi>
             </afi-safis>
           </global>
         </bgp>
       </control-plane-protocol>
     </control-plane-protocols>
   </routing>
```

A.2. Neighbor Address Family Configuration

This example shows how to configure a BGP neighbor, where the remote address is 192.0.2.1, the remote AS number is 64497, and the address family of the neighbor is IPv4 unicast. The neighbor is configured for route flap prevention and it set up for standard and large communities. In addition, BFD is configured at a neighbor level with a local multiplier of 2, a desired minimum transmit interval, and a required minimum receive interval of 3.3 ms.

```
   [note: '\' line wrapping for formatting only]

   <!--
       This example shows a neighbor configuration with damping.
   -->

   <?xml version="1.0" encoding="UTF-8"?>
   <routing
       xmlns="urn:ietf:params:xml:ns:yang:ietf-routing"
```

```
       xmlns:bt="urn:ietf:params:xml:ns:yang:iana-bgp-types"
       xmlns:bct="urn:ietf:params:xml:ns:yang:iana-bgp-community-types"\
   >
     <control-plane-protocols>
       <control-plane-protocol>
         <type
             xmlns:bgp="urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</\
   type>
         <name>name:BGP</name>
         <bgp
             xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
           <global>
             <as>64496</as>
             <afi-safis>
               <afi-safi>
                 <name>bt:ipv4-unicast</name>
               </afi-safi>
             </afi-safis>
           </global>
           <neighbors>
             <neighbor>
               <remote-address>192.0.2.1</remote-address>
               <peer-as>64497</peer-as>
               <route-flap-damping>
                 <enable>true</enable>
                 <suppress-above>4.0</suppress-above>
                 <reuse-above>3.0</reuse-above>
                 <max-flap>15.0</max-flap>
                 <reach-decay>100</reach-decay>
                 <unreach-decay>500</unreach-decay>
                 <keep-history>1000</keep-history>
               </route-flap-damping>
               <send-community>bct:standard</send-community>
               <send-community>bct:large</send-community>
               <description>"Peer Router B"</description>
               <afi-safis>
                 <afi-safi>
                   <name>bt:ipv4-unicast</name>
                 </afi-safi>
               </afi-safis>
               <transport>
                 <tcp-mss>1400</tcp-mss>
                 <mtu-discovery>true</mtu-discovery>
                 <bfd>
                   <enabled>true</enabled>
                   <local-multiplier>2</local-multiplier>
                   <desired-min-tx-interval>3300</desired-min-tx-interv\
   al>
```

```
                  <required-min-rx-interval>3300</required-min-rx-inte\
rval>
                </bfd>
              </transport>
            </neighbor>
          </neighbors>
        </bgp>
      </control-plane-protocol>
    </control-plane-protocols>
  </routing>
```

A.3. IPv6 Neighbor Configuration

This example shows how to configure a BGP peer, where the remote peer has a IPv6 address, uses TCP-AO to secure the session with the peer, and uses non-default timers for hold-time and keepalive.

[note: '\' line wrapping for formatting only]

```
<?xml version="1.0" encoding="UTF-8"?>
<key-chains
    xmlns="urn:ietf:params:xml:ns:yang:ietf-key-chain">
  <key-chain>
    <name>bgp-key-chain</name>
    <description>"An example of TCP-AO configuration for BGP"</descr\
iption>
    <key>
      <key-id>55</key-id>
      <lifetime>
        <send-lifetime>
          <start-date-time>2017-01-01T00:00:00Z</start-date-time>
          <end-date-time>2017-02-01T00:00:00Z</end-date-time>
        </send-lifetime>
        <accept-lifetime>
          <start-date-time>2016-12-31T23:59:55Z</start-date-time>
          <end-date-time>2017-02-01T00:00:05Z</end-date-time>
        </accept-lifetime>
      </lifetime>
      <crypto-algorithm>aes-cmac-prf-128</crypto-algorithm>
      <key-string>
        <keystring>testvector</keystring>
      </key-string>
      <authentication
          xmlns="urn:ietf:params:xml:ns:yang:ietf-tcp">
        <keychain>bgp-key-chain</keychain>
        <ao>
          <send-id>65</send-id>
          <recv-id>87</recv-id>
```

```
          </ao>
        </authentication>
      </key>
      <key>
        <key-id>56</key-id>
        <lifetime>
          <send-lifetime>
            <start-date-time>2017-01-01T00:00:00Z</start-date-time>
            <end-date-time>2017-02-01T00:00:00Z</end-date-time>
          </send-lifetime>
          <accept-lifetime>
            <start-date-time>2016-12-31T23:59:55Z</start-date-time>
            <end-date-time>2017-02-01T00:00:05Z</end-date-time>
          </accept-lifetime>
        </lifetime>
        <crypto-algorithm>aes-cmac-prf-128</crypto-algorithm>
        <key-string>
          <keystring>testvector</keystring>
        </key-string>
        <authentication
            xmlns="urn:ietf:params:xml:ns:yang:ietf-tcp">
          <keychain>bgp-key-chain</keychain>
          <ao>
            <send-id>65</send-id>
            <recv-id>87</recv-id>
          </ao>
        </authentication>
      </key>
    </key-chain>
  </key-chains>

  <routing
      xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
    <control-plane-protocols>
      <control-plane-protocol>
        <type
            xmlns:bgp="urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</\
type>
        <name>name:BGP</name>
        <bgp
            xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
          <global>
            <as>64496</as>
            <afi-safis>
              <afi-safi>
                <name
                    xmlns:bt=
                    "urn:ietf:params:xml:ns:yang:iana-bgp-types">bt:ip\
```

```
   v6-unicast</name>
             </afi-safi>
           </afi-safis>
         </global>
         <neighbors>
           <neighbor>
             <remote-address>2001:db8::</remote-address>
             <enabled>true</enabled>
             <transport>
               <secure-session>
                 <enabled>true</enabled>
                 <options>
                   <ao-keychain>bgp-key-chain</ao-keychain>
                 </options>
               </secure-session>
             </transport>
             <peer-as>64497</peer-as>
             <description>"Peer Router B"</description>
             <timers>
               <hold-time>120</hold-time>
               <keepalive>70</keepalive>
             </timers>
             <afi-safis>
               <afi-safi>
                 <name
                     xmlns:bt=
                     "urn:ietf:params:xml:ns:yang:iana-bgp-types">bt:\
   ipv6-unicast</name>
               </afi-safi>
             </afi-safis>
           </neighbor>
         </neighbors>
       </bgp>
     </control-plane-protocol>
   </control-plane-protocols>
 </routing>
```

A.4. VRF Configuration

This example shows how BGP can be configured for two VRFs, red and blue. In this case, the two network instances share a common AS, and distinguish between the instances using the router id.

```
   [note: '\' line wrapping for formatting only]

   <?xml version="1.0" encoding="UTF-8"?>
   <network-instances
       xmlns="urn:ietf:params:xml:ns:yang:ietf-network-instance">
     <network-instance>
       <name>vrf-red</name>
       <vrf-root>
         <routing
             xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
           <router-id>192.0.2.1</router-id>
           <control-plane-protocols>
             <control-plane-protocol>
               <type
                   xmlns:bgp=
                   "urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</type\
   >
               <name>BGP</name>
               <bgp
                   xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
                 <global>
                   <as>64496</as>
                   <afi-safis>
                     <afi-safi>
                       <name
                           xmlns:bt=
                           "urn:ietf:params:xml:ns:yang:iana-bgp-types"\
   >bt:ipv4-unicast</name>
                     </afi-safi>
                   </afi-safis>
                 </global>
               </bgp>
             </control-plane-protocol>
           </control-plane-protocols>
         </routing>
       </vrf-root>
     </network-instance>
     <network-instance>
       <name>vrf-blue</name>
       <vrf-root>
         <routing
             xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
           <router-id>192.0.2.2</router-id>
           <control-plane-protocols>
             <control-plane-protocol>
               <type
                   xmlns:bgp=
                   "urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</type\
```

```
   >
               <name>BGP</name>
               <bgp
                   xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
                 <global>
                   <as>64496</as>
                   <afi-safis>
                     <afi-safi>
                       <name
                           xmlns:bt=
                           "urn:ietf:params:xml:ns:yang:iana-bgp-types"\
   >bt:ipv4-unicast</name>
                     </afi-safi>
                   </afi-safis>
                 </global>
               </bgp>
             </control-plane-protocol>
           </control-plane-protocols>
         </routing>
       </vrf-root>
     </network-instance>
   </network-instances>
```

A.5. BGP Policy - Match Prefix and Set Community

Routing policy using community value involves configuring rules to match community values in the inbound or outbound direction. In this example, which is heavily borrowed from the example on the Cisco community page, we look at "match community exact" match, which happens only when BGP updates have the same community values as specified in the community list.

The topology in this example consists of three routers, R1, R2, and R3, configured with AS value of 1, 2 and 3 respectively. R1 advertises 5 prefixes to R2 and R3, as shown below.

* 1.1.1.1/32 and 2.2.2.2/32 with community 11:11

* 3.3.3.3/32 and 4.4.4.4/32 with community 11:11 and 22:22

* 5.5.5.5/32 with community 33.33

Route Policy TO_R2 defines the policy that R1 uses in route updates towards R2. It consists of three statements, statement 10 that has a exact match rule for the prefix list L0andL1, and a set-community action of add for 11:11. The second statement, statement 20, consists of an exact match rule for prefix list L2andL3, with a set community action of remove for 11:11 22:22. The final statement, statement 30, consists of an exact match rule for prefix list L4, with a set community action of replace for 33:33.

[note: '\' line wrapping for formatting only]

```
<?xml version="1.0" encoding="UTF-8"?>
<routing-policy
    xmlns="urn:ietf:params:xml:ns:yang:ietf-routing-policy">
  <defined-sets>
    <prefix-sets>
      <prefix-set>
        <name>L0andL1</name>
        <mode>ipv4</mode>
        <prefixes>
          <prefix-list>
            <ip-prefix>1.1.1.1/32</ip-prefix>
            <mask-length-lower>32</mask-length-lower>
            <mask-length-upper>32</mask-length-upper>
          </prefix-list>
          <prefix-list>
            <ip-prefix>2.2.2.2/32</ip-prefix>
            <mask-length-lower>32</mask-length-lower>
            <mask-length-upper>32</mask-length-upper>
          </prefix-list>
        </prefixes>
      </prefix-set>
      <prefix-set>
        <name>L2andL3</name>
        <mode>ipv4</mode>
        <prefixes>
          <prefix-list>
            <ip-prefix>3.3.3.3/32</ip-prefix>
            <mask-length-lower>32</mask-length-lower>
            <mask-length-upper>32</mask-length-upper>
          </prefix-list>
          <prefix-list>
            <ip-prefix>4.4.4.4/32</ip-prefix>
            <mask-length-lower>32</mask-length-lower>
            <mask-length-upper>32</mask-length-upper>
          </prefix-list>
        </prefixes>
      </prefix-set>
```

```
        <prefix-set>
          <name>L4</name>
          <mode>ipv4</mode>
          <prefixes>
            <prefix-list>
              <ip-prefix>5.5.5.5/32</ip-prefix>
              <mask-length-lower>32</mask-length-lower>
              <mask-length-upper>32</mask-length-upper>
            </prefix-list>
          </prefixes>
        </prefix-set>
      </prefix-sets>
    </defined-sets>
    <policy-definitions>
      <policy-definition>
        <name>TO_R2</name>
        <statements>
          <statement>
            <name>10</name>
            <conditions>
              <match-prefix-set>
                <prefix-set>L0andL1</prefix-set>
              </match-prefix-set>
            </conditions>
            <actions>
              <bgp-actions
                  xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp-policy">\

                <set-community>
                  <options>add</options>
                  <communities>11:11</communities>
                </set-community>
              </bgp-actions>
            </actions>
          </statement>
          <statement>
            <name>20</name>
            <conditions>
              <match-prefix-set>
                <prefix-set>L2andL3</prefix-set>
              </match-prefix-set>
            </conditions>
            <actions>
              <bgp-actions
                  xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp-policy">\

                <set-community>
                  <options>remove</options>
```

```
                  <communities>11:11</communities>
                  <communities>22:22</communities>
                </set-community>
              </bgp-actions>
            </actions>
          </statement>
          <statement>
            <name>30</name>
            <conditions>
              <match-prefix-set>
                <prefix-set>L4</prefix-set>
              </match-prefix-set>
            </conditions>
            <actions>
              <bgp-actions
                  xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp-policy">\

                <set-community>
                  <options>replace</options>
                  <communities>33:33</communities>
                </set-community>
              </bgp-actions>
            </actions>
          </statement>
        </statements>
      </policy-definition>
    </policy-definitions>
  </routing-policy>

  <routing
      xmlns="urn:ietf:params:xml:ns:yang:ietf-routing"
      xmlns:bt="urn:ietf:params:xml:ns:yang:iana-bgp-types"
      xmlns:bct="urn:ietf:params:xml:ns:yang:iana-bgp-community-types"\
  >
    <control-plane-protocols>
      <control-plane-protocol>
        <type
            xmlns:bgp="urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</\
  type>
        <name>BGP</name>
        <bgp
            xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
          <global>
            <as>1</as>
            <afi-safis>
              <afi-safi>
                <name>bt:ipv4-unicast</name>
              </afi-safi>
```

```
            </afi-safis>
          </global>
          <neighbors>
            <neighbor>
              <remote-address>10.1.1.2</remote-address>
              <peer-as>2</peer-as>
              <afi-safis>
                <afi-safi>
                  <name>bt:ipv4-unicast</name>
                </afi-safi>
              </afi-safis>
              <send-community>bct:standard</send-community>
              <apply-policy>
                <export-policy>TO_R2</export-policy>
                <default-export-policy>accept-route</default-export-po\
licy>
              </apply-policy>
            </neighbor>
          </neighbors>
        </bgp>
      </control-plane-protocol>
    </control-plane-protocols>
  </routing>
```

A.6. BGP Policy - Match Next-hop and Set Community

In this example, a "next-hop-set" is configured using option "invert" from "matct-set-options" to match next-hop not equal to "192.0.2.2" and then set community to 55:55.

```
   [note: '\' line wrapping for formatting only]

   <?xml version='1.0' encoding='UTF-8'?>
     <routing-policy xmlns="urn:ietf:params:xml:ns:yang:ietf-routing-po\
   licy">
       <defined-sets>
         <bgp-defined-sets xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp-\
   policy">
           <next-hop-sets>
             <next-hop-set>
               <name>nexthop_not2</name>
               <next-hop>192.0.2.2</next-hop>
             </next-hop-set>
           </next-hop-sets>
         </bgp-defined-sets>
       </defined-sets>
       <policy-definitions>
         <policy-definition>
           <name>nexthop_not2_community</name>
           <statements>
             <statement>
               <name>100</name>
               <conditions>
                 <bgp-conditions xmlns="urn:ietf:params:xml:ns:yang:iet\
   f-bgp-policy">
                   <match-next-hop-set>
                     <next-hop-set>nexthop_not2</next-hop-set>
                     <match-set-options>invert</match-set-options>
                   </match-next-hop-set>
                 </bgp-conditions>
               </conditions>
               <actions>
                 <bgp-actions xmlns="urn:ietf:params:xml:ns:yang:ietf-b\
   gp-policy">
                   <set-community>
                     <options>add</options>
                     <communities>55:55</communities>                  \

                   </set-community>
                 </bgp-actions>
               </actions>
             </statement>
           </statements>
         </policy-definition>
       </policy-definitions>
     </routing-policy>
```

Appendix B. How to add a new AFI and Augment a Module

This section explains how a new AFI can be defined in a new module and how that module can then be augmented. Assume that the new AFI being defined is called 'foo' which extends the base identity of 'afi-safi-type', and the augmentation is to add a new container for 'foo' under two different XPaths. The example shows how the base identity can be extended to add this new AFI, and then use the augmented containers be used to add 'foo' specific information.

```
module example-newafi-bgp {
  yang-version 1.1;
  namespace "http://example.com/ns/example-newafi-bgp";
  prefix example-newafi-bgp;

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

  import ietf-routing {
    prefix rt;
    reference
      "RFC 8349, A YANG Data Model for Routing Management
       (NMDA Version)";
  }

  import ietf-bgp {
    prefix "bgp";
    reference
      "RFC XXXX: YANG module for Border Gateway Protocol (BGP).";
  }

  import iana-bgp-types {
    prefix "bt";
  }

  organization
    "Newafi model group.";

  contact
    "abc@newafi.com";
  description
    "This YANG module defines and uses new AFI.";

  revision 2023-07-05 {
    description
```

```
        "Creating new AFI and using in this model";

      reference
        "RFC XXXX: YANG module for Border Gateway Protocol (BGP).";
    }

    identity foo {
      base bt:afi-safi-type;
      description
        "New AFI type foo.";
    }

    augment "/rt:routing/rt:control-plane-protocols/" +
            "rt:control-plane-protocol/bgp:bgp/bgp:global/" +
            "bgp:afi-safis/bgp:afi-safi" {
      when "derived-from-or-self(bgp:name, 'foo')" {
        description
          "This augmentation is valid for a AFI/SAFI instance
           of 'foo'";
      }
      container foo {
        description
          "Container to add 'foo' specific AFI/SAFI information.
           First add the common stuff.";
        uses bgp:mp-all-afi-safi-common;
      }
    }

    augment "/rt:routing/rt:control-plane-protocols/" +
            "rt:control-plane-protocol/bgp:bgp/" +
            "bgp:rib/bgp:afi-safis/bgp:afi-safi" {
      when "derived-from-or-self(bgp:name, 'foo')" {
        description
          "This augmentation is valid for a AFI/SAFI instance
           of 'foo'";
      }

      container foo {
        description
          "Container to add 'foo' rib specific information.
           First add the common stuff.";
        container loc-rib {
          config false;
          description
            "Container for the 'foo' BGP LOC-RIB data.";
          container routes {
            description
              "Enclosing container for list of routes in the routing
```

```
            table.";
          list route {
            key "prefix origin path-id";
            description
              "List of routes in the table, keyed by the route
               prefix, the route origin, and path-id.  The route
               origin can be either the neighbor address from which
               the route was learned, or the source protocol that
               injected the route.  The path-id distinguishes routes
               for the same prefix received from a neighbor (e.g.,
               if add-paths is enabled).";
            leaf prefix {
              type inet:ip-address;
              description
                "The 'foo' prefix corresponding to the route.";
            }
            uses bgp:bgp-loc-rib-common-keys;
            uses bgp:bgp-loc-rib-common-attr-refs;
            uses bgp:bgp-common-route-annotations-state;
            uses bgp:bgp-unknown-attr-top;
            uses bgp:rib-ext-route-annotations;
          }
          uses bgp:clear-routes;
        }
      }

      container neighbors {
        config false;
        description
          "Enclosing container for neighbor list.";
        list neighbor {
          key "neighbor-address";
          description
            "List of neighbors (peers) of the local BGP speaker.";
          leaf neighbor-address {
            type inet:ip-address;
            description
              "IP address of the BGP neighbor or peer.";
          }
          container adj-rib-in-pre {
            description
              "Per-neighbor table containing the NLRI updates
               received from the neighbor before any local input
               policy rules or filters have been applied.  This can
               be considered the 'raw' updates from the neighbor.";
            uses bgp:ipv4-adj-rib-common;
          }
          container adj-rib-in-post {
```

```
            description
              "Per-neighbor table containing the paths received from
               the neighbor that are eligible for best-path selection
               after local input policy rules have been applied.";
            uses bgp:ipv4-adj-rib-in-post;
          }
          container adj-rib-out-pre {
            description
              "Per-neighbor table containing paths eligible for
               sending (advertising) to the neighbor before output
               policy rules have been applied.";
            uses bgp:ipv4-adj-rib-common;
          }
          container adj-rib-out-post {
            description
              "Per-neighbor table containing paths eligible for
               sending (advertising) to the neighbor after output
               policy rules have been applied.";
            uses bgp:ipv4-adj-rib-common;
          }
        }
      }
    }
  }
```

Appendix C. How to deviate a module

This example shows how the BGP can be deviated to indicate two nodes that the particular implementation is choosing not to support.

```
   module example-newco-bgp {
     yang-version 1.1;
     namespace "http://example.com/ns/example-newco-bgp";
     prefix example-newco-bgp;

     import ietf-bgp {
       prefix "bgp";
     }

     organization
       "Newco model group.";

     contact
       "abc@newco.com";
     description
       "This YANG module deviates IETF BGP YANG module.";

     revision 2023-07-05 {
       description
         "Creating NewCo deviations to ietf-bgp model";

       reference
         "RFC XXXX: YANG module for Border Gateway Protocol (BGP).";
     }

     deviation "/bgp:bgp/bgp:global/bgp:graceful-restart/" +
               "bgp:restart-time" {
       deviate not-supported;
     }

     deviation "/bgp:bgp/bgp:global/bgp:graceful-restart/" +
               "bgp:stale-route-time" {
       deviate not-supported;
     }
   }
```

Appendix D. Complete configuration tree diagram

Here is a complete tree diagram for the configuration and operational part of the model.

```
   module: ietf-bgp

     augment /rt:routing/rt:control-plane-protocols
               /rt:control-plane-protocol:
       +--rw bgp
          +--rw global!
          |  +--rw as                          inet:as-number
```

```
|  +--rw identifier?                yang:dotted-quad
|  +--rw distance
|  |  +--rw external?   uint8
|  |  +--rw internal?   uint8
|  +--rw confederation
|  |  +--rw enabled?      boolean
|  |  +--rw identifier?   inet:as-number
|  |  +--rw member-as*    inet:as-number
|  +--rw graceful-restart {bt:graceful-restart}?
|  |  +--rw enabled?             boolean
|  |  +--rw restart-time?        bt:graceful-restart-time-type
|  |  +--rw stale-routes-time?   uint32
|  |  +--rw helper-only?         boolean
|  +--rw use-multiple-paths
|  |  +--rw enabled?   boolean
|  |  +--rw ebgp
|  |  |  +--rw allow-multiple-as?   boolean
|  |  |  +--rw maximum-paths?       uint32
|  |  +--rw ibgp
|  |     +--rw maximum-paths?   uint32
|  +--rw route-selection-options
|  |  +--rw always-compare-med?           boolean
|  |  +--rw ignore-as-path-length?        boolean
|  |  +--rw external-compare-router-id?   boolean
|  |  +--rw advertise-inactive-routes?    boolean
|  |  +--rw enable-aigp?                  boolean
|  |  +--rw ignore-next-hop-igp-metric?   boolean
|  |  +--rw enable-med?                   boolean
|  |  +--rw med-plus-igp
|  |     +--rw enabled?          boolean
|  |     +--rw igp-multiplier?   uint16
|  |     +--rw med-multiplier?   uint16
|  +--rw afi-safis
|  |  +--rw afi-safi* [name]
|  |     +--rw name                       identityref
|  |     +--rw enabled?                   boolean
|  |     +--ro statistics
|  |     |  +--ro total-paths?      yang:gauge32
|  |     |  +--ro total-prefixes?   yang:gauge32
|  |     +--rw graceful-restart {bt:graceful-restart}?
|  |     |  +--rw enabled?   boolean
|  |     +--rw route-selection-options
|  |     |  +--rw always-compare-med?           boolean
|  |     |  +--rw ignore-as-path-length?        boolean
|  |     |  +--rw external-compare-router-id?   boolean
|  |     |  +--rw advertise-inactive-routes?    boolean
|  |     |  +--rw enable-aigp?                  boolean
|  |     |  +--rw ignore-next-hop-igp-metric?   boolean
```

```
|  |  |  |  +--rw enable-med?                    boolean
|  |  |  |  +--rw med-plus-igp
|  |  |  |     +--rw enabled?          boolean
|  |  |  |     +--rw igp-multiplier?   uint16
|  |  |  |     +--rw med-multiplier?   uint16
|  |  |  +--rw add-paths {bt:add-paths}?
|  |  |  |  +--rw receive?                   boolean
|  |  |  |  +--rw (send)?
|  |  |  |  |  +--:(max)
|  |  |  |  |  |  +--rw max?                uint8
|  |  |  |  |  +--:(all)
|  |  |  |  |     +--rw all?                empty
|  |  |  |  +--rw eligible-prefix-policy?   leafref
|  |  |  +--rw use-multiple-paths
|  |  |  |  +--rw enabled?   boolean
|  |  |  |  +--rw ebgp
|  |  |  |  |  +--rw allow-multiple-as?   boolean
|  |  |  |  |  +--rw maximum-paths?       uint32
|  |  |  |  +--rw ibgp
|  |  |  |     +--rw maximum-paths?   uint32
|  |  |  +--rw apply-policy
|  |  |  |  +--rw import-policy*           leafref
|  |  |  |  +--rw default-import-policy?   default-policy-type
|  |  |  |  +--rw export-policy*           leafref
|  |  |  |  +--rw default-export-policy?   default-policy-type
|  |  |  +--rw ipv4-unicast
|  |  |  |  +--rw prefix-limit
|  |  |  |  |  +--rw max-prefixes?            uint32
|  |  |  |  |  +--rw warning-threshold-pct?
|  |  |  |  |  |       rt-types:percentage
|  |  |  |  |  +--rw teardown?                boolean
|  |  |  |  |  +--rw idle-time?               union
|  |  |  |  |  +--ro prefix-limit-exceeded?   boolean
|  |  |  |  +--rw send-default-route?   boolean
|  |  |  +--rw ipv6-unicast
|  |  |  |  +--rw prefix-limit
|  |  |  |  |  +--rw max-prefixes?            uint32
|  |  |  |  |  +--rw warning-threshold-pct?
|  |  |  |  |  |       rt-types:percentage
|  |  |  |  |  +--rw teardown?                boolean
|  |  |  |  |  +--rw idle-time?               union
|  |  |  |  |  +--ro prefix-limit-exceeded?   boolean
|  |  |  |  +--rw send-default-route?   boolean
|  |  |  +--rw ipv4-labeled-unicast
|  |  |  |  +--rw prefix-limit
|  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |     +--rw warning-threshold-pct?
|  |  |  |     |       rt-types:percentage
```

```
|  |  |  |     +--rw teardown?                boolean
|  |  |  |     +--rw idle-time?               union
|  |  |  |     +--ro prefix-limit-exceeded?   boolean
|  |  |  +--rw ipv6-labeled-unicast
|  |  |  |  +--rw prefix-limit
|  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |     +--rw warning-threshold-pct?
|  |  |  |     |       rt-types:percentage
|  |  |  |     +--rw teardown?                boolean
|  |  |  |     +--rw idle-time?               union
|  |  |  |     +--ro prefix-limit-exceeded?   boolean
|  |  |  +--rw l3vpn-ipv4-unicast
|  |  |  |  +--rw prefix-limit
|  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |     +--rw warning-threshold-pct?
|  |  |  |     |       rt-types:percentage
|  |  |  |     +--rw teardown?                boolean
|  |  |  |     +--rw idle-time?               union
|  |  |  |     +--ro prefix-limit-exceeded?   boolean
|  |  |  +--rw l3vpn-ipv6-unicast
|  |  |  |  +--rw prefix-limit
|  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |     +--rw warning-threshold-pct?
|  |  |  |     |       rt-types:percentage
|  |  |  |     +--rw teardown?                boolean
|  |  |  |     +--rw idle-time?               union
|  |  |  |     +--ro prefix-limit-exceeded?   boolean
|  |  |  +--rw l3vpn-ipv4-multicast
|  |  |  |  +--rw prefix-limit
|  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |     +--rw warning-threshold-pct?
|  |  |  |     |       rt-types:percentage
|  |  |  |     +--rw teardown?                boolean
|  |  |  |     +--rw idle-time?               union
|  |  |  |     +--ro prefix-limit-exceeded?   boolean
|  |  |  +--rw l3vpn-ipv6-multicast
|  |  |  |  +--rw prefix-limit
|  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |     +--rw warning-threshold-pct?
|  |  |  |     |       rt-types:percentage
|  |  |  |     +--rw teardown?                boolean
|  |  |  |     +--rw idle-time?               union
|  |  |  |     +--ro prefix-limit-exceeded?   boolean
|  |  |  +--rw l2vpn-vpls
|  |  |  |  +--rw prefix-limit
|  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |     +--rw warning-threshold-pct?
|  |  |  |     |       rt-types:percentage
```

```
|  |  |     |     +--rw teardown?                boolean
|  |  |     |     +--rw idle-time?               union
|  |  |     |     +--ro prefix-limit-exceeded?   boolean
|  |  |     +--rw l2vpn-evpn
|  |  |        +--rw prefix-limit
|  |  |           +--rw max-prefixes?            uint32
|  |  |           +--rw warning-threshold-pct?
|  |  |           |       rt-types:percentage
|  |  |           +--rw teardown?                boolean
|  |  |           +--rw idle-time?               union
|  |  |           +--ro prefix-limit-exceeded?   boolean
|  |  +--rw apply-policy
|  |  |  +--rw import-policy*           leafref
|  |  |  +--rw default-import-policy?   default-policy-type
|  |  |  +--rw export-policy*           leafref
|  |  |  +--rw default-export-policy?   default-policy-type
|  |  +--ro statistics
|  |     +--ro total-paths?      yang:gauge32
|  |     +--ro total-prefixes?   yang:gauge32
|  +--rw neighbors
|  |  +--rw neighbor* [remote-address]
|  |  |  +--rw remote-address            inet:ip-address
|  |  |  +--rw peer-group?
|  |  |  |       -> ../../../peer-groups/peer-group/name
|  |  |  +--ro local-address?            inet:ip-address
|  |  |  +--ro local-port?               inet:port-number
|  |  |  +--ro remote-port?              inet:port-number
|  |  |  +--ro peer-type?                bt:peer-type
|  |  |  +--ro identifier?               yang:dotted-quad
|  |  |  +--ro dynamically-configured?   empty
|  |  |  +--rw enabled?                  boolean
|  |  |  +--rw peer-as?                  inet:as-number
|  |  |  +--rw local-as?                 inet:as-number
|  |  |  +--rw remove-private-as?
|  |  |  |       bt:remove-private-as-option
|  |  |  +--rw route-flap-damping {bt:damping}?
|  |  |  |  +--rw enable?           boolean
|  |  |  |  +--rw suppress-above?   decimal64
|  |  |  |  +--rw reuse-above?      decimal64
|  |  |  |  +--rw max-flap?         decimal64
|  |  |  |  +--rw reach-decay?      uint32
|  |  |  |  +--rw unreach-decay?    uint32
|  |  |  |  +--rw keep-history?     uint32
|  |  |  +--rw send-community*           identityref
|  |  |  |       {bct:send-communities}?
|  |  |  +--rw description?              string
|  |  |  +--rw timers
|  |  |  |  +--rw connect-retry-interval?            uint16
```

```
|  |  |  +--rw hold-time?                            uint16
|  |  |  +--ro negotiated-hold-time?                 uint16
|  |  |  +--rw keepalive?                            uint16
|  |  |  +--rw min-as-origination-interval?          uint16
|  |  |  +--rw min-route-advertisement-interval?     uint16
|  |  +--rw transport
|  |  |  +--rw local-address?    union
|  |  |  +--rw tcp-mss?          tcp:mss
|  |  |  +--rw mtu-discovery?    boolean
|  |  |  +--rw ebgp-multihop
|  |  |  |  +--rw enabled?        boolean
|  |  |  |  +--rw multihop-ttl?   uint8
|  |  |  +--rw passive-mode?     boolean
|  |  |  +--rw ttl-security?     uint8 {bt:ttl-security}?
|  |  |  +--rw secure-session
|  |  |  |  +--rw enabled?   boolean
|  |  |  |  +--rw options
|  |  |  |     +--rw (option)?
|  |  |  |        +--:(ao)
|  |  |  |        |  +--rw ao-keychain?
|  |  |  |        |           key-chain:key-chain-ref
|  |  |  |        +--:(md5)
|  |  |  |        |  +--rw md5-keychain?
|  |  |  |        |           key-chain:key-chain-ref
|  |  |  |        +--:(ipsec)
|  |  |  |           +--rw sa?             string
|  |  |  +--rw bfd {bt:bfd}?
|  |  |     +--rw enabled?                           boolean
|  |  |     +--rw local-multiplier?                  multiplier
|  |  |     |       {client-base-cfg-parms}?
|  |  |     +--rw (interval-config-type)?
|  |  |             {client-base-cfg-parms}?
|  |  |        +--:(tx-rx-intervals)
|  |  |        |  +--rw desired-min-tx-interval?    uint32
|  |  |        |  +--rw required-min-rx-interval?   uint32
|  |  |        +--:(single-interval)
|  |  |                 {single-minimum-interval}?
|  |  |           +--rw min-interval?                uint32
|  |  +--rw treat-as-withdraw?        boolean
|  |  +--rw logging-options
|  |  |  +--rw log-neighbor-state-changes?   boolean
|  |  +--rw route-reflector
|  |  |  +--rw cluster-id?   bt:rr-cluster-id-type
|  |  |  +--rw client?       boolean
|  |  +--rw as-path-options
|  |  |  +--rw allow-own-as?             uint8
|  |  |  +--rw replace-peer-as?          boolean
|  |  |  +--rw disable-peer-as-filter?   boolean
```

```
|   |   +--rw add-paths {bt:add-paths}?
|   |   |  +--rw receive?                  boolean
|   |   |  +--rw (send)?
|   |   |  |  +--:(max)
|   |   |  |  |  +--rw max?                uint8
|   |   |  |  +--:(all)
|   |   |  |     +--rw all?                empty
|   |   |  +--rw eligible-prefix-policy?   leafref
|   |   +--rw use-multiple-paths
|   |   |  +--rw enabled?   boolean
|   |   |  +--rw ebgp
|   |   |     +--rw allow-multiple-as?   boolean
|   |   +--rw apply-policy
|   |   |  +--rw import-policy*           leafref
|   |   |  +--rw default-import-policy?   default-policy-type
|   |   |  +--rw export-policy*           leafref
|   |   |  +--rw default-export-policy?   default-policy-type
|   |   +--rw graceful-restart {bt:graceful-restart}?
|   |   |  +--rw enabled?             boolean
|   |   |  +--rw restart-time?
|   |   |  |       bt:graceful-restart-time-type
|   |   |  +--rw stale-routes-time?   uint32
|   |   |  +--rw helper-only?         boolean
|   |   |  +--ro peer-restart-time?   uint16
|   |   |  +--ro peer-restarting?     boolean
|   |   |  +--ro local-restarting?    boolean
|   |   |  +--ro mode?                enumeration
|   |   +--rw prefix-limit
|   |   |  +--rw max-prefixes?            uint32
|   |   |  +--rw warning-threshold-pct?   rt-types:percentage
|   |   |  +--rw teardown?                boolean
|   |   |  +--rw idle-time?               union
|   |   |  +--ro prefix-limit-exceeded?   boolean
|   |   +--rw afi-safis
|   |   |  +--rw afi-safi* [name]
|   |   |     +--rw name                    identityref
|   |   |     +--rw enabled?                boolean
|   |   |     +--ro active?                 boolean
|   |   |     +--ro prefixes
|   |   |     |  +--ro received?    yang:zero-based-counter32
|   |   |     |  +--ro sent?        yang:zero-based-counter32
|   |   |     |  +--ro installed?   yang:gauge32
|   |   |     +--rw graceful-restart {bt:graceful-restart}?
|   |   |     |  +--rw enabled?
|   |   |     |  |       boolean
|   |   |     |  +--ro received?
|   |   |     |  |       boolean
|   |   |     |  +--ro advertised?
```

```
|  |  |  |  |  |       boolean
|  |  |  |  |  +--ro local-forwarding-state-preserved?
|  |  |  |  |  |       boolean
|  |  |  |  |  +--ro forwarding-state-preserved?
|  |  |  |  |  |       boolean
|  |  |  |  |  +--ro end-of-rib-received?
|  |  |  |  |             boolean
|  |  |  |  +--rw apply-policy
|  |  |  |  |  +--rw import-policy*           leafref
|  |  |  |  |  +--rw default-import-policy?
|  |  |  |  |  |       default-policy-type
|  |  |  |  |  +--rw export-policy*           leafref
|  |  |  |  |  +--rw default-export-policy?
|  |  |  |  |             default-policy-type
|  |  |  |  +--rw ipv4-unicast
|  |  |  |  |  +--rw prefix-limit
|  |  |  |  |  |  +--rw max-prefixes?            uint32
|  |  |  |  |  |  +--rw warning-threshold-pct?
|  |  |  |  |  |  |       rt-types:percentage
|  |  |  |  |  |  +--rw teardown?                boolean
|  |  |  |  |  |  +--rw idle-time?               union
|  |  |  |  |  |  +--ro prefix-limit-exceeded?   boolean
|  |  |  |  |  +--rw send-default-route?   boolean
|  |  |  |  +--rw ipv6-unicast
|  |  |  |  |  +--rw prefix-limit
|  |  |  |  |  |  +--rw max-prefixes?            uint32
|  |  |  |  |  |  +--rw warning-threshold-pct?
|  |  |  |  |  |  |       rt-types:percentage
|  |  |  |  |  |  +--rw teardown?                boolean
|  |  |  |  |  |  +--rw idle-time?               union
|  |  |  |  |  |  +--ro prefix-limit-exceeded?   boolean
|  |  |  |  |  +--rw send-default-route?   boolean
|  |  |  |  +--rw ipv4-labeled-unicast
|  |  |  |  |  +--rw prefix-limit
|  |  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |  |     +--rw warning-threshold-pct?
|  |  |  |  |     |       rt-types:percentage
|  |  |  |  |     +--rw teardown?                boolean
|  |  |  |  |     +--rw idle-time?               union
|  |  |  |  |     +--ro prefix-limit-exceeded?   boolean
|  |  |  |  +--rw ipv6-labeled-unicast
|  |  |  |  |  +--rw prefix-limit
|  |  |  |  |     +--rw max-prefixes?            uint32
|  |  |  |  |     +--rw warning-threshold-pct?
|  |  |  |  |     |       rt-types:percentage
|  |  |  |  |     +--rw teardown?                boolean
|  |  |  |  |     +--rw idle-time?               union
|  |  |  |  |     +--ro prefix-limit-exceeded?   boolean
```

```
|  |  |  |     +--rw l3vpn-ipv4-unicast
|  |  |  |     |  +--rw prefix-limit
|  |  |  |     |     +--rw max-prefixes?            uint32
|  |  |  |     |     +--rw warning-threshold-pct?
|  |  |  |     |     |       rt-types:percentage
|  |  |  |     |     +--rw teardown?                boolean
|  |  |  |     |     +--rw idle-time?               union
|  |  |  |     |     +--ro prefix-limit-exceeded?   boolean
|  |  |  |     +--rw l3vpn-ipv6-unicast
|  |  |  |     |  +--rw prefix-limit
|  |  |  |     |     +--rw max-prefixes?            uint32
|  |  |  |     |     +--rw warning-threshold-pct?
|  |  |  |     |     |       rt-types:percentage
|  |  |  |     |     +--rw teardown?                boolean
|  |  |  |     |     +--rw idle-time?               union
|  |  |  |     |     +--ro prefix-limit-exceeded?   boolean
|  |  |  |     +--rw l3vpn-ipv4-multicast
|  |  |  |     |  +--rw prefix-limit
|  |  |  |     |     +--rw max-prefixes?            uint32
|  |  |  |     |     +--rw warning-threshold-pct?
|  |  |  |     |     |       rt-types:percentage
|  |  |  |     |     +--rw teardown?                boolean
|  |  |  |     |     +--rw idle-time?               union
|  |  |  |     |     +--ro prefix-limit-exceeded?   boolean
|  |  |  |     +--rw l3vpn-ipv6-multicast
|  |  |  |     |  +--rw prefix-limit
|  |  |  |     |     +--rw max-prefixes?            uint32
|  |  |  |     |     +--rw warning-threshold-pct?
|  |  |  |     |     |       rt-types:percentage
|  |  |  |     |     +--rw teardown?                boolean
|  |  |  |     |     +--rw idle-time?               union
|  |  |  |     |     +--ro prefix-limit-exceeded?   boolean
|  |  |  |     +--rw l2vpn-vpls
|  |  |  |     |  +--rw prefix-limit
|  |  |  |     |     +--rw max-prefixes?            uint32
|  |  |  |     |     +--rw warning-threshold-pct?
|  |  |  |     |     |       rt-types:percentage
|  |  |  |     |     +--rw teardown?                boolean
|  |  |  |     |     +--rw idle-time?               union
|  |  |  |     |     +--ro prefix-limit-exceeded?   boolean
|  |  |  |     +--rw l2vpn-evpn
|  |  |  |     |  +--rw prefix-limit
|  |  |  |     |     +--rw max-prefixes?            uint32
|  |  |  |     |     +--rw warning-threshold-pct?
|  |  |  |     |     |       rt-types:percentage
|  |  |  |     |     +--rw teardown?                boolean
|  |  |  |     |     +--rw idle-time?               union
|  |  |  |     |     +--ro prefix-limit-exceeded?   boolean
```

```
|  |  |  |     +--rw use-multiple-paths
|  |  |  |        +--rw enabled?   boolean
|  |  |  |        +--rw ebgp
|  |  |  |           +--rw allow-multiple-as?   boolean
|  |  |  +--rw session-state?            enumeration
|  |  |  +--ro last-established?         yang:date-and-time
|  |  |  +--ro capabilities
|  |  |  |  +--ro advertised-capabilities* [code index]
|  |  |  |  |  +--ro code     uint8
|  |  |  |  |  +--ro index    uint8
|  |  |  |  |  +--ro name?    identityref
|  |  |  |  |  +--ro value
|  |  |  |  |     +--ro mpbgp
|  |  |  |  |     |  +--ro afi?    iana-rt-types:address-family
|  |  |  |  |     |  +--ro safi?   iana-rt-types:bgp-safi
|  |  |  |  |     |  +--ro name?   identityref
|  |  |  |  |     +--ro graceful-restart
|  |  |  |  |     |  +--ro flags?
|  |  |  |  |     |  |       bt:graceful-restart-flags
|  |  |  |  |     |  +--ro unknown-flags?   bits
|  |  |  |  |     |  +--ro restart-time?
|  |  |  |  |     |  |       bt:graceful-restart-time-type
|  |  |  |  |     |  +--ro afi-safis* []
|  |  |  |  |     |     +--ro afi?
|  |  |  |  |     |     |       iana-rt-types:address-family
|  |  |  |  |     |     +--ro safi?
|  |  |  |  |     |     |       iana-rt-types:bgp-safi
|  |  |  |  |     |     +--ro afi-safi-flags?
|  |  |  |  |     |     |       bt:graceful-restart-flags-for-afi
|  |  |  |  |     |     +--ro afi-safi-unknown-flags?   bits
|  |  |  |  |     +--ro asn32
|  |  |  |  |     |  +--ro as?   inet:as-number
|  |  |  |  |     +--ro add-paths
|  |  |  |  |        +--ro afi-safis* []
|  |  |  |  |           +--ro afi?
|  |  |  |  |           |       iana-rt-types:address-family
|  |  |  |  |           +--ro safi?   iana-rt-types:bgp-safi
|  |  |  |  |           +--ro mode?   enumeration
|  |  |  |  +--ro received-capabilities* [code index]
|  |  |  |  |  +--ro code     uint8
|  |  |  |  |  +--ro index    uint8
|  |  |  |  |  +--ro name?    identityref
|  |  |  |  |  +--ro value
|  |  |  |  |     +--ro mpbgp
|  |  |  |  |     |  +--ro afi?    iana-rt-types:address-family
|  |  |  |  |     |  +--ro safi?   iana-rt-types:bgp-safi
|  |  |  |  |     |  +--ro name?   identityref
|  |  |  |  |     +--ro graceful-restart
```

```
|  |  |  |  |  |  +--ro flags?
|  |  |  |  |  |  |       bt:graceful-restart-flags
|  |  |  |  |  |  +--ro unknown-flags?   bits
|  |  |  |  |  |  +--ro restart-time?
|  |  |  |  |  |  |       bt:graceful-restart-time-type
|  |  |  |  |  |  +--ro afi-safis* []
|  |  |  |  |  |     +--ro afi?
|  |  |  |  |  |     |       iana-rt-types:address-family
|  |  |  |  |  |     +--ro safi?
|  |  |  |  |  |     |       iana-rt-types:bgp-safi
|  |  |  |  |  |     +--ro afi-safi-flags?
|  |  |  |  |  |     |       bt:graceful-restart-flags-for-afi
|  |  |  |  |  |     +--ro afi-safi-unknown-flags?   bits
|  |  |  |  |  +--ro asn32
|  |  |  |  |  |  +--ro as?   inet:as-number
|  |  |  |  |  +--ro add-paths
|  |  |  |  |     +--ro afi-safis* []
|  |  |  |  |        +--ro afi?
|  |  |  |  |        |       iana-rt-types:address-family
|  |  |  |  |        +--ro safi?   iana-rt-types:bgp-safi
|  |  |  |  |        +--ro mode?   enumeration
|  |  |  |  +--ro negotiated-capabilities*   identityref
|  |  |  +--ro errors
|  |  |  |  +--ro received
|  |  |  |  |  +--ro last-notification?
|  |  |  |  |  |       yang:date-and-time
|  |  |  |  |  +--ro last-error?
|  |  |  |  |  |       identityref
|  |  |  |  |  +--ro last-error-code?                    uint8
|  |  |  |  |  +--ro last-error-subcode?                 uint8
|  |  |  |  |  +--ro last-encapsulated-error?
|  |  |  |  |  |       identityref
|  |  |  |  |  +--ro last-encapsulated-error-code?       uint8
|  |  |  |  |  +--ro last-encapsulated-error-subcode?    uint8
|  |  |  |  |  +--ro last-error-data?                    binary
|  |  |  |  +--ro sent
|  |  |  |     +--ro last-notification?
|  |  |  |     |       yang:date-and-time
|  |  |  |     +--ro last-error?
|  |  |  |     |       identityref
|  |  |  |     +--ro last-error-code?                    uint8
|  |  |  |     +--ro last-error-subcode?                 uint8
|  |  |  |     +--ro last-encapsulated-error?
|  |  |  |     |       identityref
|  |  |  |     +--ro last-encapsulated-error-code?       uint8
|  |  |  |     +--ro last-encapsulated-error-subcode?    uint8
|  |  |  |     +--ro last-error-data?                    binary
|  |  |  +--ro statistics
```

```
|  |     +--ro established-transitions?
|  |     |       yang:zero-based-counter32
|  |     +--ro messages
|  |     |  +--ro total-received?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro total-sent?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro updates-received?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro updates-sent?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro erroneous-updates-withdrawn?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro erroneous-updates-attribute-discarded?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro in-update-elapsed-time?
|  |     |  |       yang:gauge32
|  |     |  +--ro notifications-received?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro notifications-sent?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro route-refreshes-received?
|  |     |  |       yang:zero-based-counter32
|  |     |  +--ro route-refreshes-sent?
|  |     |          yang:zero-based-counter32
|  |     +--ro queues
|  |     |  +--ro input?    yang:gauge32
|  |     |  +--ro output?   yang:gauge32
|  |     +---x clear {bt:clear-statistics}?
|  |        +---w input
|  |        |  +---w clear-at?   yang:date-and-time
|  |        +--ro output
|  |           +--ro clear-finished-at?   yang:date-and-time
|  +---n established
|  |  +-- remote-address?   -> ../../neighbor/remote-address
|  +---n backward-transition
|  |  +-- remote-addr?
|  |  |       -> ../../neighbor/remote-address
|  |  +-- notification-received
|  |  |  +-- last-notification?
|  |  |  |       yang:date-and-time
|  |  |  +-- last-error?                          identityref
|  |  |  +-- last-error-code?                     uint8
|  |  |  +-- last-error-subcode?                  uint8
|  |  |  +-- last-encapsulated-error?             identityref
|  |  |  +-- last-encapsulated-error-code?        uint8
|  |  |  +-- last-encapsulated-error-subcode?     uint8
|  |  +-- notification-sent
```

```
|  |     +-- last-notification?
|  |     |       yang:date-and-time
|  |     +-- last-error?                          identityref
|  |     +-- last-error-code?                     uint8
|  |     +-- last-error-subcode?                  uint8
|  |     +-- last-encapsulated-error?             identityref
|  |     +-- last-encapsulated-error-code?        uint8
|  |     +-- last-encapsulated-error-subcode?     uint8
|  +---x clear {bt:clear-neighbors}?
|     +---w input
|     |  +---w (operation)?
|     |  |  +--:(operation-admin)
|     |  |  |  +---w admin?          empty
|     |  |  +--:(operation-hard)
|     |  |  |  +---w hard?           empty
|     |  |  +--:(operation-soft)
|     |  |  |  +---w soft?           empty
|     |  |  +--:(operation-soft-inbound)
|     |  |     +---w soft-inbound?   empty {bt:route-refresh}?
|     |  +---w clear-at?             yang:date-and-time
|     +--ro output
|        +--ro clear-finished-at?   yang:date-and-time
+--rw peer-groups
|  +--rw peer-group* [name]
|     +--rw name                    string
|     +--rw peer-as?                inet:as-number
|     +--rw local-as?               inet:as-number
|     +--rw remove-private-as?      bt:remove-private-as-option
|     +--rw route-flap-damping {bt:damping}?
|     |  +--rw enable?              boolean
|     |  +--rw suppress-above?      decimal64
|     |  +--rw reuse-above?         decimal64
|     |  +--rw max-flap?            decimal64
|     |  +--rw reach-decay?         uint32
|     |  +--rw unreach-decay?       uint32
|     |  +--rw keep-history?        uint32
|     +--rw send-community*         identityref
|     |       {bct:send-communities}?
|     +--rw description?            string
|     +--rw timers
|     |  +--rw connect-retry-interval?             uint16
|     |  +--rw hold-time?                          uint16
|     |  +--ro negotiated-hold-time?               uint16
|     |  +--rw keepalive?                          uint16
|     |  +--rw min-as-origination-interval?        uint16
|     |  +--rw min-route-advertisement-interval?   uint16
|     +--rw transport
|     |  +--rw local-address?    union
```

```
|      |  +--rw tcp-mss?          tcp:mss
|      |  +--rw mtu-discovery?    boolean
|      |  +--rw ebgp-multihop
|      |  |  +--rw enabled?        boolean
|      |  |  +--rw multihop-ttl?   uint8
|      |  +--rw passive-mode?     boolean
|      |  +--rw ttl-security?     uint8 {bt:ttl-security}?
|      |  +--rw secure-session
|      |  |  +--rw enabled?   boolean
|      |  |  +--rw options
|      |  |     +--rw (option)?
|      |  |        +--:(ao)
|      |  |        |  +--rw ao-keychain?
|      |  |        |          key-chain:key-chain-ref
|      |  |        +--:(md5)
|      |  |        |  +--rw md5-keychain?
|      |  |        |          key-chain:key-chain-ref
|      |  |        +--:(ipsec)
|      |  |           +--rw sa?             string
|      |  +--rw bfd {bt:bfd}?
|      |     +--rw enabled?                          boolean
|      |     +--rw local-multiplier?                 multiplier
|      |     |       {client-base-cfg-parms}?
|      |     +--rw (interval-config-type)?
|      |              {client-base-cfg-parms}?
|      |        +--:(tx-rx-intervals)
|      |        |  +--rw desired-min-tx-interval?    uint32
|      |        |  +--rw required-min-rx-interval?   uint32
|      |        +--:(single-interval)
|      |                 {single-minimum-interval}?
|      |           +--rw min-interval?               uint32
|      +--rw treat-as-withdraw?    boolean
|      +--rw logging-options
|      |  +--rw log-neighbor-state-changes?   boolean
|      +--rw route-reflector
|      |  +--rw cluster-id?   bt:rr-cluster-id-type
|      |  +--rw client?       boolean
|      +--rw as-path-options
|      |  +--rw allow-own-as?             uint8
|      |  +--rw replace-peer-as?          boolean
|      |  +--rw disable-peer-as-filter?   boolean
|      +--rw add-paths {bt:add-paths}?
|      |  +--rw receive?                  boolean
|      |  +--rw (send)?
|      |  |  +--:(max)
|      |  |  |  +--rw max?               uint8
|      |  |  +--:(all)
|      |  |     +--rw all?               empty
```

```
|      |  +--rw eligible-prefix-policy?   leafref
|      +--rw use-multiple-paths
|      |  +--rw enabled?   boolean
|      |  +--rw ebgp
|      |     +--rw allow-multiple-as?   boolean
|      +--rw apply-policy
|      |  +--rw import-policy*           leafref
|      |  +--rw default-import-policy?   default-policy-type
|      |  +--rw export-policy*           leafref
|      |  +--rw default-export-policy?   default-policy-type
|      +--rw dynamic-peers
|      |  +--rw dynamic-peer-list* [prefix]
|      |     +--rw prefix    inet:ip-prefix
|      +--rw graceful-restart {bt:graceful-restart}?
|      |  +--rw enabled?             boolean
|      |  +--rw restart-time?
|      |  |       bt:graceful-restart-time-type
|      |  +--rw stale-routes-time?   uint32
|      |  +--rw helper-only?         boolean
|      +--rw prefix-limit
|      |  +--rw max-prefixes?            uint32
|      |  +--rw warning-threshold-pct?   rt-types:percentage
|      |  +--rw teardown?                boolean
|      |  +--rw idle-time?               union
|      +--rw afi-safis
|         +--rw afi-safi* [name]
|            +--rw name                    identityref
|            +--rw enabled?                boolean
|            +--rw graceful-restart {bt:graceful-restart}?
|            |  +--rw enabled?   boolean
|            +--rw add-paths {bt:add-paths}?
|            |  +--rw receive?                  boolean
|            |  +--rw (send)?
|            |  |  +--:(max)
|            |  |  |  +--rw max?               uint8
|            |  |  +--:(all)
|            |  |     +--rw all?               empty
|            |  +--rw eligible-prefix-policy?   leafref
|            +--rw use-multiple-paths
|            |  +--rw enabled?   boolean
|            |  +--rw ebgp
|            |     +--rw allow-multiple-as?   boolean
|            +--rw apply-policy
|            |  +--rw import-policy*           leafref
|            |  +--rw default-import-policy?
|            |  |       default-policy-type
|            |  +--rw export-policy*           leafref
|            |  +--rw default-export-policy?
```

```
|           |          default-policy-type
|           +--rw ipv4-unicast
|           |  +--rw prefix-limit
|           |  |  +--rw max-prefixes?            uint32
|           |  |  +--rw warning-threshold-pct?
|           |  |  |       rt-types:percentage
|           |  |  +--rw teardown?                boolean
|           |  |  +--rw idle-time?               union
|           |  |  +--ro prefix-limit-exceeded?   boolean
|           |  +--rw send-default-route?   boolean
|           +--rw ipv6-unicast
|           |  +--rw prefix-limit
|           |  |  +--rw max-prefixes?            uint32
|           |  |  +--rw warning-threshold-pct?
|           |  |  |       rt-types:percentage
|           |  |  +--rw teardown?                boolean
|           |  |  +--rw idle-time?               union
|           |  |  +--ro prefix-limit-exceeded?   boolean
|           |  +--rw send-default-route?   boolean
|           +--rw ipv4-labeled-unicast
|           |  +--rw prefix-limit
|           |     +--rw max-prefixes?            uint32
|           |     +--rw warning-threshold-pct?
|           |     |       rt-types:percentage
|           |     +--rw teardown?                boolean
|           |     +--rw idle-time?               union
|           |     +--ro prefix-limit-exceeded?   boolean
|           +--rw ipv6-labeled-unicast
|           |  +--rw prefix-limit
|           |     +--rw max-prefixes?            uint32
|           |     +--rw warning-threshold-pct?
|           |     |       rt-types:percentage
|           |     +--rw teardown?                boolean
|           |     +--rw idle-time?               union
|           |     +--ro prefix-limit-exceeded?   boolean
|           +--rw l3vpn-ipv4-unicast
|           |  +--rw prefix-limit
|           |     +--rw max-prefixes?            uint32
|           |     +--rw warning-threshold-pct?
|           |     |       rt-types:percentage
|           |     +--rw teardown?                boolean
|           |     +--rw idle-time?               union
|           |     +--ro prefix-limit-exceeded?   boolean
|           +--rw l3vpn-ipv6-unicast
|           |  +--rw prefix-limit
|           |     +--rw max-prefixes?            uint32
|           |     +--rw warning-threshold-pct?
|           |     |       rt-types:percentage
```

```
|            |     +--rw teardown?                boolean
|            |     +--rw idle-time?               union
|            |     +--ro prefix-limit-exceeded?   boolean
|            +--rw l3vpn-ipv4-multicast
|            |  +--rw prefix-limit
|            |     +--rw max-prefixes?            uint32
|            |     +--rw warning-threshold-pct?
|            |     |       rt-types:percentage
|            |     +--rw teardown?                boolean
|            |     +--rw idle-time?               union
|            |     +--ro prefix-limit-exceeded?   boolean
|            +--rw l3vpn-ipv6-multicast
|            |  +--rw prefix-limit
|            |     +--rw max-prefixes?            uint32
|            |     +--rw warning-threshold-pct?
|            |     |       rt-types:percentage
|            |     +--rw teardown?                boolean
|            |     +--rw idle-time?               union
|            |     +--ro prefix-limit-exceeded?   boolean
|            +--rw l2vpn-vpls
|            |  +--rw prefix-limit
|            |     +--rw max-prefixes?            uint32
|            |     +--rw warning-threshold-pct?
|            |     |       rt-types:percentage
|            |     +--rw teardown?                boolean
|            |     +--rw idle-time?               union
|            |     +--ro prefix-limit-exceeded?   boolean
|            +--rw l2vpn-evpn
|               +--rw prefix-limit
|                  +--rw max-prefixes?            uint32
|                  +--rw warning-threshold-pct?
|                  |       rt-types:percentage
|                  +--rw teardown?                boolean
|                  +--rw idle-time?               union
|                  +--ro prefix-limit-exceeded?   boolean
+--ro rib
   +--ro attr-sets
   |  +--ro attr-set* [index]
   |     +--ro index         uint64
   |     +--ro attributes
   |        +--ro origin?
   |        |       bt:bgp-origin-attr-type
   |        +--ro as-path
   |        |  +--ro segment* []
   |        |     +--ro type?     identityref
   |        |     +--ro member*   inet:as-number
   |        +--ro next-hop?              inet:ip-address
   |        +--ro link-local-next-hop?   inet:ipv6-address
```

```
|        +--ro med?                    uint32
|        +--ro local-pref?             uint32
|        +--ro as4-path
|        |  +--ro segment* []
|        |     +--ro type?     identityref
|        |     +--ro member*   inet:as-number
|        +--ro aggregator
|        |  +--ro as?           inet:as-number
|        |  +--ro identifier?   yang:dotted-quad
|        +--ro aggregator4
|        |  +--ro as4?          inet:as-number
|        |  +--ro identifier?   yang:dotted-quad
|        +--ro atomic-aggregate?      boolean
|        +--ro originator-id?         yang:dotted-quad
|        +--ro cluster-list*          yang:dotted-quad
|        +--ro aigp-metric?           uint64
+--ro communities
|  +--ro community* [index]
|     +--ro index        uint64
|     +--ro community*   union
+--ro ext-communities
|  +--ro ext-community* [index]
|     +--ro index                uint64
|     +--ro ext-community*       bct:bgp-ext-community-type
|     +--ro ext-community-raw*   string
+--ro ipv6-ext-communities
|  +--ro ipv6-ext-community* [index]
|     +--ro index                        uint64
|     +--ro ipv6-ext-community*
|     |       bct:bgp-ipv6-ext-community-type
|     +--ro ipv6-ext-community-raw*   string
+--ro large-communities
|  +--ro large-community* [index]
|     +--ro index              uint64
|     +--ro large-community*   bct:bgp-large-community-type
+--ro afi-safis
   +--ro afi-safi* [name]
      +--ro name            identityref
      +--ro ipv4-unicast
      |  +--ro loc-rib
      |  |  +--ro routes
      |  |     +--ro route* [prefix origin path-id]
      |  |        +--ro prefix
      |  |        |       inet:ipv4-prefix
      |  |        +--ro origin                   union
      |  |        +--ro path-id                  uint32
      |  |        +--ro attr-index?              leafref
      |  |        +--ro community-index?         leafref
```

```
|  |        +--ro ext-community-index?     leafref
|  |        +--ro large-community-index?   leafref
|  |        +--ro last-modified?
|  |        |       yang:timeticks
|  |        +--ro eligible-route?          boolean
|  |        +--ro ineligible-reason?
|  |        |       identityref
|  |        +--ro unknown-attributes
|  |        |  +--ro unknown-attribute* [attr-type]
|  |        |     +--ro optional?     boolean
|  |        |     +--ro transitive?   boolean
|  |        |     +--ro partial?      boolean
|  |        |     +--ro extended?     boolean
|  |        |     +--ro attr-type     uint8
|  |        |     +--ro attr-len?     uint16
|  |        |     +--ro attr-value?   binary
|  |        +--ro reject-reason?           union
|  +--ro neighbors
|     +--ro neighbor* [neighbor-address]
|        +--ro neighbor-address    inet:ip-address
|        +--ro adj-rib-in-pre
|        |  +--ro routes
|        |  |  +--ro route* [prefix path-id]
|        |  |     +--ro prefix
|        |  |     |       inet:ipv4-prefix
|        |  |     +--ro path-id
|        |  |     |       uint32
|        |  |     +--ro attr-index?
|        |  |     |       leafref
|        |  |     +--ro community-index?
|        |  |     |       leafref
|        |  |     +--ro ext-community-index?
|        |  |     |       leafref
|        |  |     +--ro large-community-index?
|        |  |     |       leafref
|        |  |     +--ro last-modified?
|        |  |     |       yang:timeticks
|        |  |     +--ro eligible-route?
|        |  |     |       boolean
|        |  |     +--ro ineligible-reason?
|        |  |     |       identityref
|        |  |     +--ro unknown-attributes
|        |  |     |  +--ro unknown-attribute*
|        |  |     |           [attr-type]
|        |  |     |     +--ro optional?     boolean
|        |  |     |     +--ro transitive?   boolean
|        |  |     |     +--ro partial?      boolean
|        |  |     |     +--ro extended?     boolean
```

```
|        |  |     |     +--ro attr-type     uint8
|        |  |     |     +--ro attr-len?     uint16
|        |  |     |     +--ro attr-value?   binary
|        |  |     +--ro reject-reason?
|        |  |               union
|        |  +--ro clear-routes {bt:clear-routes}?
|        |     +---x clear
|        |        +---w input
|        |        |  +---w clear-at?
|        |        |          yang:date-and-time
|        |        +--ro output
|        |           +--ro clear-finished-at?
|        |                     yang:date-and-time
|        +--ro adj-rib-in-post
|        |  +--ro routes
|        |  |  +--ro route* [prefix path-id]
|        |  |     +--ro prefix
|        |  |     |       inet:ipv4-prefix
|        |  |     +--ro path-id
|        |  |     |       uint32
|        |  |     +--ro attr-index?
|        |  |     |       leafref
|        |  |     +--ro community-index?
|        |  |     |       leafref
|        |  |     +--ro ext-community-index?
|        |  |     |       leafref
|        |  |     +--ro large-community-index?
|        |  |     |       leafref
|        |  |     +--ro last-modified?
|        |  |     |       yang:timeticks
|        |  |     +--ro eligible-route?
|        |  |     |       boolean
|        |  |     +--ro ineligible-reason?
|        |  |     |       identityref
|        |  |     +--ro best-path?
|        |  |     |       boolean
|        |  |     +--ro unknown-attributes
|        |  |     |  +--ro unknown-attribute*
|        |  |     |          [attr-type]
|        |  |     |     +--ro optional?     boolean
|        |  |     |     +--ro transitive?   boolean
|        |  |     |     +--ro partial?      boolean
|        |  |     |     +--ro extended?     boolean
|        |  |     |     +--ro attr-type     uint8
|        |  |     |     +--ro attr-len?     uint16
|        |  |     |     +--ro attr-value?   binary
|        |  |     +--ro reject-reason?
|        |  |               union
```

```
|          |  +--ro clear-routes {bt:clear-routes}?
|          |     +---x clear
|          |        +---w input
|          |        |  +---w clear-at?
|          |        |          yang:date-and-time
|          |        +--ro output
|          |           +--ro clear-finished-at?
|          |                   yang:date-and-time
|          +--ro adj-rib-out-pre
|          |  +--ro routes
|          |  |  +--ro route* [prefix path-id]
|          |  |     +--ro prefix
|          |  |     |       inet:ipv4-prefix
|          |  |     +--ro path-id
|          |  |     |       uint32
|          |  |     +--ro attr-index?
|          |  |     |       leafref
|          |  |     +--ro community-index?
|          |  |     |       leafref
|          |  |     +--ro ext-community-index?
|          |  |     |       leafref
|          |  |     +--ro large-community-index?
|          |  |     |       leafref
|          |  |     +--ro last-modified?
|          |  |     |       yang:timeticks
|          |  |     +--ro eligible-route?
|          |  |     |       boolean
|          |  |     +--ro ineligible-reason?
|          |  |     |       identityref
|          |  |     +--ro unknown-attributes
|          |  |     |  +--ro unknown-attribute*
|          |  |     |          [attr-type]
|          |  |     |     +--ro optional?     boolean
|          |  |     |     +--ro transitive?   boolean
|          |  |     |     +--ro partial?      boolean
|          |  |     |     +--ro extended?     boolean
|          |  |     |     +--ro attr-type     uint8
|          |  |     |     +--ro attr-len?     uint16
|          |  |     |     +--ro attr-value?   binary
|          |  |     +--ro reject-reason?
|          |  |             union
|          |  +--ro clear-routes {bt:clear-routes}?
|          |     +---x clear
|          |        +---w input
|          |        |  +---w clear-at?
|          |        |          yang:date-and-time
|          |        +--ro output
|          |           +--ro clear-finished-at?
```

```
                  |        |                   yang:date-and-time
                  |        +--ro adj-rib-out-post
                  |           +--ro routes
                  |           |  +--ro route* [prefix path-id]
                  |           |     +--ro prefix
                  |           |     |       inet:ipv4-prefix
                  |           |     +--ro path-id
                  |           |     |       uint32
                  |           |     +--ro attr-index?
                  |           |     |       leafref
                  |           |     +--ro community-index?
                  |           |     |       leafref
                  |           |     +--ro ext-community-index?
                  |           |     |       leafref
                  |           |     +--ro large-community-index?
                  |           |     |       leafref
                  |           |     +--ro last-modified?
                  |           |     |       yang:timeticks
                  |           |     +--ro eligible-route?
                  |           |     |       boolean
                  |           |     +--ro ineligible-reason?
                  |           |     |       identityref
                  |           |     +--ro unknown-attributes
                  |           |     |  +--ro unknown-attribute*
                  |           |     |          [attr-type]
                  |           |     |     +--ro optional?     boolean
                  |           |     |     +--ro transitive?   boolean
                  |           |     |     +--ro partial?      boolean
                  |           |     |     +--ro extended?     boolean
                  |           |     |     +--ro attr-type     uint8
                  |           |     |     +--ro attr-len?     uint16
                  |           |     |     +--ro attr-value?   binary
                  |           |     +--ro reject-reason?
                  |           |             union
                  |           +--ro clear-routes {bt:clear-routes}?
                  |              +---x clear
                  |                 +---w input
                  |                 |  +---w clear-at?
                  |                 |          yang:date-and-time
                  |                 +--ro output
                  |                    +--ro clear-finished-at?
                  |                            yang:date-and-time
                  |
                 +--ro ipv6-unicast
                    +--ro loc-rib
                    |  +--ro routes
                    |     +--ro route* [prefix origin path-id]
                    |        +--ro prefix
                    |        |       inet:ipv6-prefix
```

```
|        +--ro origin                    union
|        +--ro path-id                   uint32
|        +--ro attr-index?               leafref
|        +--ro community-index?          leafref
|        +--ro ext-community-index?      leafref
|        +--ro large-community-index?    leafref
|        +--ro last-modified?
|        |       yang:timeticks
|        +--ro eligible-route?           boolean
|        +--ro ineligible-reason?
|        |       identityref
|        +--ro unknown-attributes
|        |  +--ro unknown-attribute* [attr-type]
|        |     +--ro optional?     boolean
|        |     +--ro transitive?   boolean
|        |     +--ro partial?      boolean
|        |     +--ro extended?     boolean
|        |     +--ro attr-type     uint8
|        |     +--ro attr-len?     uint16
|        |     +--ro attr-value?   binary
|        +--ro reject-reason?            union
|     +--ro neighbors
|        +--ro neighbor* [neighbor-address]
|           +--ro neighbor-address    inet:ip-address
|           +--ro adj-rib-in-pre
|           |  +--ro routes
|           |  |  +--ro route* [prefix path-id]
|           |  |     +--ro prefix
|           |  |     |       inet:ipv6-prefix
|           |  |     +--ro path-id
|           |  |     |       uint32
|           |  |     +--ro attr-index?
|           |  |     |       leafref
|           |  |     +--ro community-index?
|           |  |     |       leafref
|           |  |     +--ro ext-community-index?
|           |  |     |       leafref
|           |  |     +--ro large-community-index?
|           |  |     |       leafref
|           |  |     +--ro last-modified?
|           |  |     |       yang:timeticks
|           |  |     +--ro eligible-route?
|           |  |     |       boolean
|           |  |     +--ro ineligible-reason?
|           |  |     |       identityref
|           |  |     +--ro unknown-attributes
|           |  |     |  +--ro unknown-attribute*
|           |  |     |          [attr-type]
```

```
|  |  |     +--ro optional?     boolean
|  |  |     +--ro transitive?   boolean
|  |  |     +--ro partial?      boolean
|  |  |     +--ro extended?     boolean
|  |  |     +--ro attr-type     uint8
|  |  |     +--ro attr-len?     uint16
|  |  |     +--ro attr-value?   binary
|  |  +--ro reject-reason?
|  |           union
|  +--ro clear-routes {bt:clear-routes}?
|     +---x clear
|        +---w input
|        |  +---w clear-at?
|        |           yang:date-and-time
|        +--ro output
|           +--ro clear-finished-at?
|                    yang:date-and-time
+--ro adj-rib-in-post
|  +--ro routes
|  |  +--ro route* [prefix path-id]
|  |     +--ro prefix
|  |     |       inet:ipv6-prefix
|  |     +--ro path-id
|  |     |       uint32
|  |     +--ro attr-index?
|  |     |       leafref
|  |     +--ro community-index?
|  |     |       leafref
|  |     +--ro ext-community-index?
|  |     |       leafref
|  |     +--ro large-community-index?
|  |     |       leafref
|  |     +--ro last-modified?
|  |     |       yang:timeticks
|  |     +--ro eligible-route?
|  |     |       boolean
|  |     +--ro ineligible-reason?
|  |     |       identityref
|  |     +--ro best-path?
|  |     |       boolean
|  |     +--ro unknown-attributes
|  |     |  +--ro unknown-attribute*
|  |     |           [attr-type]
|  |     |     +--ro optional?     boolean
|  |     |     +--ro transitive?   boolean
|  |     |     +--ro partial?      boolean
|  |     |     +--ro extended?     boolean
|  |     |     +--ro attr-type     uint8
```

```
|  |  |     +--ro attr-len?     uint16
|  |  |     +--ro attr-value?   binary
|  |     +--ro reject-reason?
|  |               union
|  +--ro clear-routes {bt:clear-routes}?
|     +---x clear
|        +---w input
|        |  +---w clear-at?
|        |           yang:date-and-time
|        +--ro output
|           +--ro clear-finished-at?
|                    yang:date-and-time
+--ro adj-rib-out-pre
|  +--ro routes
|  |  +--ro route* [prefix path-id]
|  |     +--ro prefix
|  |     |       inet:ipv6-prefix
|  |     +--ro path-id
|  |     |       uint32
|  |     +--ro attr-index?
|  |     |       leafref
|  |     +--ro community-index?
|  |     |       leafref
|  |     +--ro ext-community-index?
|  |     |       leafref
|  |     +--ro large-community-index?
|  |     |       leafref
|  |     +--ro last-modified?
|  |     |       yang:timeticks
|  |     +--ro eligible-route?
|  |     |       boolean
|  |     +--ro ineligible-reason?
|  |     |       identityref
|  |     +--ro unknown-attributes
|  |     |  +--ro unknown-attribute*
|  |     |           [attr-type]
|  |     |     +--ro optional?     boolean
|  |     |     +--ro transitive?   boolean
|  |     |     +--ro partial?      boolean
|  |     |     +--ro extended?     boolean
|  |     |     +--ro attr-type     uint8
|  |     |     +--ro attr-len?     uint16
|  |     |     +--ro attr-value?   binary
|  |     +--ro reject-reason?
|  |               union
|  +--ro clear-routes {bt:clear-routes}?
|     +---x clear
|        +---w input
```

```
|         |  +---w clear-at?
|         |          yang:date-and-time
|         +--ro output
|            +--ro clear-finished-at?
|                    yang:date-and-time
+--ro adj-rib-out-post
   +--ro routes
   |  +--ro route* [prefix path-id]
   |     +--ro prefix
   |     |       inet:ipv6-prefix
   |     +--ro path-id
   |     |       uint32
   |     +--ro attr-index?
   |     |       leafref
   |     +--ro community-index?
   |     |       leafref
   |     +--ro ext-community-index?
   |     |       leafref
   |     +--ro large-community-index?
   |     |       leafref
   |     +--ro last-modified?
   |     |       yang:timeticks
   |     +--ro eligible-route?
   |     |       boolean
   |     +--ro ineligible-reason?
   |     |       identityref
   |     +--ro unknown-attributes
   |     |  +--ro unknown-attribute*
   |     |          [attr-type]
   |     |     +--ro optional?     boolean
   |     |     +--ro transitive?   boolean
   |     |     +--ro partial?      boolean
   |     |     +--ro extended?     boolean
   |     |     +--ro attr-type     uint8
   |     |     +--ro attr-len?     uint16
   |     |     +--ro attr-value?   binary
   |     +--ro reject-reason?
   |             union
   +--ro clear-routes {bt:clear-routes}?
      +---x clear
         +---w input
         |  +---w clear-at?
         |          yang:date-and-time
         +--ro output
            +--ro clear-finished-at?
                    yang:date-and-time
```

Appendix E. Complete policy tree diagram

Here is a complete tree diagram for the policy portion of the model.

```
module: ietf-bgp-policy

  augment /rt-pol:routing-policy/rt-pol:defined-sets:
    +--rw bgp-defined-sets
       +--rw as-path-sets
       |  +--rw as-path-set* [name]
       |     +--rw name      string
       |     +--rw member*   string
       +--rw community-sets
       |  +--rw community-set* [name]
       |     +--rw name      string
       |     +--rw member*   union
       +--rw ext-community-sets
       |  +--rw ext-community-set* [name]
       |     +--rw name      string
       |     +--rw member*   union
       +--rw ipv6-ext-community-sets
       |  +--rw ipv6-ext-community-set* [name]
       |     +--rw name      string
       |     +--rw member*   union
       +--rw large-community-sets
       |  +--rw large-community-set* [name]
       |     +--rw name      string
       |     +--rw member*   union
       +--rw next-hop-sets
          +--rw next-hop-set* [name]
             +--rw name        string
             +--rw next-hop*   bgp-next-hop-type
  augment /rt-pol:routing-policy/rt-pol:policy-definitions
            /rt-pol:policy-definition/rt-pol:statements
            /rt-pol:statement/rt-pol:conditions:
    +--rw bgp-conditions
       +--rw local-pref
       |  +--rw value?            uint32
       |  +--rw (operation)?
       |     +--:(eq)
       |     |  +--rw eq?         empty
       |     +--:(lt-or-eq)
       |     |  +--rw lt-or-eq?   empty
       |     +--:(gt-or-eq)
       |        +--rw gt-or-eq?   empty
       +--rw med
       |  +--rw value?            uint32
       |  +--rw (operation)?
```

```
|      +--:(eq)
|      |  +--rw eq?          empty
|      +--:(lt-or-eq)
|      |  +--rw lt-or-eq?    empty
|      +--:(gt-or-eq)
|         +--rw gt-or-eq?    empty
+--rw origin-eq?                          bt:bgp-origin-attr-type
+--rw match-afi-safi
|  +--rw afi-safi-in*          identityref
|  +--rw match-set-options?    match-set-options-type
+--rw match-neighbor
|  +--rw neighbor-eq*          inet:ip-address
|  +--rw match-set-options?    match-set-options-type
+--rw route-type?                         enumeration
+--rw community-count
|  +--rw community-count?    uint32
|  +--rw (operation)?
|     +--:(eq)
|     |  +--rw eq?           empty
|     +--:(lt-or-eq)
|     |  +--rw lt-or-eq?     empty
|     +--:(gt-or-eq)
|        +--rw gt-or-eq?     empty
+--rw as-path-length
|  +--rw as-path-length?    uint32
|  +--rw (operation)?
|     +--:(eq)
|     |  +--rw eq?          empty
|     +--:(lt-or-eq)
|     |  +--rw lt-or-eq?    empty
|     +--:(gt-or-eq)
|        +--rw gt-or-eq?    empty
+--rw match-community-set
|  +--rw community-set?        leafref
|  +--rw match-set-options?    match-set-options-type
+--rw match-ext-community-set
|  +--rw ext-community-set?          leafref
|  +--rw ext-community-match-kind?   enumeration
|  +--rw match-set-options?          match-set-options-type
+--rw match-ipv6-ext-community-set
|  +--rw ipv6-ext-community-set?          leafref
|  +--rw ipv6-ext-community-match-kind?   enumeration
|  +--rw match-set-options?
|           match-set-options-type
+--rw match-large-community-set
|  +--rw large-community-set?    leafref
|  +--rw match-set-options?      match-set-options-type
+--rw match-as-path-set
```

```
          |  +--rw as-path-set?         leafref
          |  +--rw match-set-options?   match-set-options-type
          +--rw match-next-hop-set
             +--rw next-hop-set?        leafref
             +--rw match-set-options?   match-set-options-type
    augment /rt-pol:routing-policy/rt-pol:policy-definitions
              /rt-pol:policy-definition/rt-pol:statements
              /rt-pol:statement/rt-pol:actions:
      +--rw bgp-actions
         +--rw set-route-origin?         bt:bgp-origin-attr-type
         +--rw set-local-pref?           uint32
         +--rw set-next-hop?             bgp-next-hop-type
         +--rw set-med?                  bgp-set-med-type
         +--rw set-as-path-prepend
         |  +--rw repeat-n?   uint8
         |  +--rw asn*        inet:as-number
         +--rw set-community
         |  +--rw options?
         |  |       bgp-set-community-option-type
         |  +--rw (method)?
         |     +--:(inline)
         |     |  +--rw communities*         union
         |     +--:(reference)
         |        +--rw community-set-ref?   leafref
         +--rw set-ext-community
         |  +--rw options?
         |  |       bgp-set-community-option-type
         |  +--rw (method)?
         |     +--:(inline)
         |     |  +--rw communities*
         |     |          bct:bgp-ext-community-type
         |     +--:(reference)
         |        +--rw ext-community-set-ref?   leafref
         +--rw set-ipv6-ext-community
         |  +--rw options?
         |  |       bgp-set-community-option-type
         |  +--rw (method)?
         |     +--:(inline)
         |     |  +--rw communities*
         |     |          bct:bgp-ipv6-ext-community-type
         |     +--:(reference)
         |        +--rw ipv6-ext-community-set-ref?   leafref
         +--rw set-large-community
            +--rw options?
            |       bgp-set-community-option-type
            +--rw (method)?
               +--:(inline)
               |  +--rw communities*
```

```
                |          bct:bgp-large-community-type
               +--:(reference)
                  +--rw large-community-set-ref?   leafref
```

Appendix F. BGP YANG Model AS_PATH Regular Expressions

F.1. AS_PATH Regular Expressions in Implementations

Implementations of BGP have different ways of rendering the received BGP AS_PATH in their interfaces. Deployed implementations generally render the AS_PATHs consistently from left to right where the left-most AS number matches the neighbor AS; most recently prepended AS, per Section 5.1.2 of [RFC4271]. Deployed implementations also use different characters to match the associated AS_PATH segment types. Those segment types are AS_SEQUENCE, AS_SET, AS_CONFED_SEQUENCE, AS_CONFED_SET. Implementations also have no fully standardized way to render white-space between AS numbers in combination with possible characters denoting the AS_PATH segment types.

This YANG model standardizes no particular rendering format for an AS_PATH. This permits the model to be generally applicable for implementations of BGP and does not conflict with deployed formatting choices.

AS_PATH regular expressions are a common tool used in BGP policy. Such regular expressions match on components of the AS_PATH. In deployed implementations of BGP, there are two common high-level forms of AS_PATH matching:

* Character based tokens. In this form, AS numbers are treated as a string of characters. Regular expressions are evaluated vs. the characters in the string. E.g. "64503+" will match "64503", "645033", etc.

* Integer based tokens. In this form, AS numbers are treated as tokens. Regular expressions are evaluated vs. the full integers. E.g. "64503+" will match "64503", "64503 64503".

F.2. Why not use standard POSIX regular expressions?

* It would be necessary to standardize the format of the AS_PATH in this module. See above for reasons this was not chosen.

* Consistently dealing with separator token in the configured AS_PATH regular expression can be error-prone. Operators interacting with BGP data using "grep" are familiar with this challenge.

* POSIX character-based regular expressions readily map to implementations that match on character based tokens, but do not easily map to integer based token implementation. Integer based token matches can be expressed in implementations that do character based tokens.

* Additionally, YANG by default doesn't use POSIX for its regular expressions.

F.3. BGP YANG AS_PATH Regular Expressions

To address the points above, the BGP YANG AS_PATH regular expressions are a subset of, and are inspired by, the prior work in Section 5.4 of RPSL [RFC2622].

ASN
: ASN is an 4-byte AS Number, from 1..4294967295. 0 is not a valid AS Number and is not permitted to be advertised in BGP AS_PATHs per [RFC7607].

.
: matches any single AS number in the AS_PATH.

[...]
: is an AS number set. It matches AS_PATHs with AS numbers listed between the brackets. The AS numbers in the set are separated by white-space characters. If a '-' is used between two AS numbers in this set, all AS numbers in the range between the two AS numbers are included in the set.

[^...]
: is a complemented AS number set. It matches an AS_PATH which is not matched by the numbers in the set.

^
: Matches the empty string at the beginning of an AS_PATH.

$
: Matches the empty string at the end of an AS_PATH.

Unary postfix operators * + ? {m} {m,n} {m,}
: For a regular expression A, A* matches zero or more occurrences of A; A+ matches one or more occurrences of A; A? matches zero or one occurrence of A; A{m} matches m occurrence of A; A{m,n} matches m to n occurrence of A; A{m,} matches m or more occurrence of A. For example, [AS1 AS2]{2} matches AS1 AS1, AS1 AS2, AS2 AS1, and AS2 AS2.

Binary catenation operator
: This is an implicit operator and exists between two regular expressions A and B when no other explicit operator is specified. The resulting expression A B matches an AS-path if A matches some prefix of the AS-path and B matches the rest of the AS-path.

Binary alternative (or) operator |
: For a regular expressions A and B, A | B matches any AS-path that is matched by A or B.

(...)
: Parenthesis can be used to override the default order of evaluation.

Whitespace in a BGP YANG AS_PATH regular expression is only used to separate tokens within the regular expression. It is otherwise without meaning and may be included or omitted for visual clarity.

F.4. Matching AS_PATH Segment Types in AS_PATH Regular Expressions

There are no deployed implementations of BGP AS_PATH regular expression matches that permit matching of AS_PATH segment types. Known implementations simply ignore the segment types for their internal representation strings used by that implementation's regular expression. While this would seem to be problematic, and may create difficulties in some specific scenarios, the following reasons have mitigated this consideration over the years:

* AS_SET or AS_CONFED_SET in the AS_PATH not only introduces additional character tokens for delimiting the segment, but mean that the contents of that position within the AS_PATH may vary. In general, implementations of AS_SETs will sort the contents of the AS_SET, but this is only a recommendation and not a protocol requirement. AS_SETs are being deprecated for implementation and deployment in BGP due to the issues that they raise in BGP security features. See [I-D.ietf-idr-deprecate-as-set-confed-set].

* BGP Confederation are the always the left-most ASes in the AS_PATH. As a result, matching on confederation ASes may be done using the '^' anchor character against the locally-configured Confederation member ASes. These are typically either publicly registered AS numbers under the control of a single entity, or private AS numbers that are scrubbed as part of route filtering purposes at the provider's network edge.

F.5. BGP YANG AS_PATH Regular Expressions ABNF

TBD...

F.6. Example BGP YANG AS_PATH Regular Expressions

64496
: Matches the AS_PATH that contains at least one instance of 64496, regardless of its position.

^64496$
: Matches the AS_PATH that is exactly one AS long, and that AS is 64496.

^64496
: Matches the AS_PATH where the left-most (neighbor) AS is 64496.

64496$
: Matches the AS_PATH where the right-most (origin) AS is 64496.

[64496 64500]
: Matches the AS_PATH where at least one of the ASes is either 64496 or 64500.

[^64496 64500]
: Matches the AS_PATH where neither 64496 nor 64500 are present in the AS_PATH.

[64496-64511]
: Matches the AS_PATH where at least one AS number is in the range 64496..64511, inclusive. This matches the documentation AS numbers in [RFC5398].

64496 64500
: Matches the AS_PATH where 64496 is present and immediately followed by 64500.

64496 | 64500
: Matches the AS_PATH where 64496 or 64500 occur in the AS_PATH.

^ 64496 .{2} (64500 | [64505-64511])
: Matches the AS_PATH where the left-most (neighbor) AS is 64496, is followed immediately by any two ASes and either 64500 or an AS in the range 64505..64511.

Authors' Addresses

Mahesh Jethanandani
Kloud Services
Email: mjethanandani@gmail.com

Keyur Patel
Arrcus
CA
United States of America
Email: keyur@arrcus.com

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI 48176
United States of America
Email: shares@ndzh.com

Jeffrey Haas
Juniper Networks
Email: jhaas@pfrc.org