BGP Model for Service Provider Networks
draft-ietf-idr-bgp-model-02

Abstract

This document defines a YANG data model for configuring and managing BGP, including protocol, policy, and operational aspects 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 http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 19, 2017.
1. Introduction

YANG [RFC6020] is a data modeling language that was introduced to define data for managing networked devices using NETCONF [RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces (e.g. ReST) and encodings other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for a number of interfaces, such as
This document describes a YANG data model for the BGP [RFC4271] protocol, including various protocol extensions, policy configuration, as well as defining key operational state data. 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.

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 [RFC4271], [RFC1997], [RFC4456], [RFC4760], [RFC3065], [RFC2439], [RFC4724], and [RFC6811].

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 [I-D.ietf-rtgwg-policy-model]. The model also supports operational state data to provide a common model for reading BGP-related state from a BGP speaker.

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.

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.

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.
- multiprotocol configuration -- configuration affecting individual address-families within BGP [RFC4760].
- neighbor configuration -- configuration affecting an individual neighbor within BGP.
- neighbor multiprotocol configuration -- configuration affecting individual address-families for a neighbor within BGP.
- policy configuration -- hooks for application of the policies defined in [I-D.ietf-rtgwg-policy-model] 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 RFC 6991 [RFC6991].

Throughout the model, the approach described in [I-D.openconfig-netmod-opstate] is used to represent configuration (intended state), operational and derived state data. That is to
say, that each container holds a "config" and "state" sub-container - with the config container being used for configurable parameters, and the state container container holding both the operational state of configurable leaves, and derived counters and statistical information.

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.

```
+--rw bgp!
   |   +-- (global-configuration-options)
   |   +--rw neighbors
   |      |   +--rw neighbor* [neighbor-address]
   |      |      +-- (neighbor-configuration-options)
   |      +--rw peer-groups
   |         +--rw peer-group* [peer-group-name]
   |         +-- (neighbor-configuration-options)
```

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 to 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 following address-families are currently supported by the model:

```
+--rw bgp!
    +--rw global
        +--rw afi-safis
            +--rw afi-safi* [afi-safi-name]
                +--rw afi-safi-name  -> ../config/afi-safi-name
                    |                    +--rw ipv4-unicast
                    |                        | ...
                    |                        +--rw ipv6-unicast
                    |                        | ...  
                    |                        +--rw ipv4-labelled-unicast
                    |                        | ...  
                    |                        +--rw ipv6-labelled-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 references the generic YANG routing policy model described in [I-D.ietf-rtgwg-policy-model], 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 specific neighbor or group.

```
+--rw bgp
  ++--rw global
  |   ++--rw afi-safi
  |     |   ++--rw afi-safi* [afi-safi-name]
  |     |   ++--rw apply-policy
  |     ++--rw apply-policy
  ++--rw neighbors
  |   ++--rw neighbor* [neighbor-address]
  |     |   ++--rw afi-safi
  |     |     |   ++--rw afi-safi* [afi-safi-name]
  |     |     |     |   ++--rw apply-policy
  |     |     ++--rw apply-policy
  ++--rw peer-groups
  |   ++--rw peer-group* [peer-group-name]
  |     |   ++--rw afi-safi
  |     |     |   ++--rw afi-safi* [afi-safi-name]
  |     |     |     |   ++--rw apply-policy
  |     |     ++--rw apply-policy
```

2.3. Operational state overview

The BGP operational model contains data which relates to the operational state of the various elements of the BGP router. As noted in Section 2 - the approach described in [I-D.openconfig-netmod-opstate] is utilized for the inclusion of operational and statistical data. To this end, the "-state" groupings (those that contain derived operational parameters) are
contained within the BGP operational model - and included within the relevant "state" containers throughout the core BGP model. In some cases, operational information may be relevant to one instance of a common grouping, but not another - for example, the number of received, advertised, and installed prefixes is relevant on a per-neighbor-basis, but is not required (or meaningful) in the peer-group context. Groupings are defined with the appropriate operational state data accordingly.

3. Relation to other YANG data models

The BGP model is intended to work within a larger framework model, such as the Network Instance model [I-D.openconfig-rtgwg-network-instance] which provides a comprehensive model for defining VRFs, associated routing protocols, multiple protocol instances, and inter-protocol and inter-instance routing policies. The current version of the model imports and instantiates the BGP model in its tree at /network-instances/network-instance/protocols/protocol/bgp/...

It is also possible to integrate the BGP model with the Routing Management model [I-D.ietf-netmod-routing-cfg] or the Network Device Organizational Model [I-D.rtgyangdt-rtgwg-device-model], both of which define the notion of routing instances, or VRFs.

4. Security Considerations

BGP configuration has a significant impact on network operations, and as such any related protocol or model carries potential security risks.

YANG data models are generally designed to be used with the NETCONF protocol over an SSH transport. This provides an authenticated and secure channel over which to transfer BGP configuration and operational data. Note that use of alternate transport or data encoding (e.g., JSON over HTTPS) would require similar mechanisms for authenticating and securing access to configuration data.

Most of the data elements in the configuration model could be considered sensitive from a security standpoint. Unauthorized access or invalid data could cause major disruption.

5. IANA Considerations

An appropriate namespace URI will be registered in the IETF XML Registry" [RFC3688]. The BGP YANG modules will be registered in the "YANG Module Names" registry [RFC6020].
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, openconfig-bgp.yang, includes the following submodules:

- openconfig-bgp-common - defines the groupings that are common across more than one context (where contexts are neighbor, group, global)
- openconfig-bgp-common-multiprotocol - defines the groupings that are common across more than one context, and relate to multiprotocol BGP
- openconfig-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)
- openconfig-bgp-global - groupings with data specific to the global context
- openconfig-bgp-peer-group - groupings with data specific to the peer group context
- openconfig-bgp-neighbor - groupings with data specific to the neighbor context

Additional modules include:

- openconfig-bgp-types - common type and identity definitions for BGP, including BGP policy
- openconfig-bgp-policy - BGP-specific policy data definitions for use with [I-D.ietf-rtgwg-policy-model] (described in more detail Section 2.2)

7. BGP main module and submodule for base items

<CODE BEGINS> file "ietf-bgp.yang"

module ietf-bgp {
    yang-version "1";

namespace "urn:ietf:params:xml:ns:yang:ietf-bgp";

prefix "bgp";

// import some basic inet types
import openconfig-extensions { prefix oc-ext; }
import ietf-routing-policy { prefix rpol; }

// Include the OpenConfig BGP submodules
// Common: defines the groupings that are common across more than
//         one context (where contexts are neighbor, group, global)
include ietf-bgp-common;

// Multiprotocol: defines the groupings that are common across more
//                than one context, and relate to Multiprotocol
include ietf-bgp-common-multiprotocol;

// Structure: defines groupings that are shared but are solely used for
//            structural reasons.
include ietf-bgp-common-structure;

// Include peer-group/neighbor/global - these define the groupings
// that are specific to one context
include ietf-bgp-peer-group;
include ietf-bgp-neighbor;
include ietf-bgp-global;

// meta
organization
    "OpenConfig working group";

contact
    "OpenConfig working group
    netopenconfig@googlegroups.com";

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 ]

oc-ext:openconfig-version "2.1.1";

revision "2016-06-21" {
  description
  "OpenConfig BGP refactor";
  reference "2.1.1";
}

revision "2016-06-06" {
  description
  "OpenConfig public release";
  reference "2.1.0";
}

revision "2016-03-31" {
  description
  "OpenConfig public release";
  reference "2.0.1";
}

grouping bgp-top {
  description
  "Top-level grouping for the BGP model data";

ccontai**ner bgp {
  description
  "Top-level configuration and state for the BGP router";

ccontai**ner global {
  description
  "Global configuration for the BGP router";
    uses bgp-global-base;
    uses rpol:apply-policy-group;
  }

ccontai**ner neighbors {
  description
  "Configuration for BGP neighbors";

uses bgp-neighbor-list;
}

container peer-groups {
  description
    "Configuration for BGP peer-groups";
  uses bgp-peer-group-list;
}

uses bgp-top;

</CODE ENDS>

<CODE BEGINS> file "ietf-bgp-common.yang"

submodule ietf-bgp-common {

  belongs-to ietf-bgp {
    prefix "bgp";
  }

  import openconfig-extensions { prefix oc-ext; }
  import ietf-bgp-types { prefix oc-bgp-types; }
  import ietf-routing-policy { prefix rpol; }
  import ietf-inet-types { prefix inet; }

  // meta
  organization
    "OpenConfig working group";

  contact
    "OpenConfig working group
    netopenconfig@googlegroups.com";

  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.";

  oc-ext:openconfig-version "2.1.1";

  revision "2016-06-21" {

description
"OpenConfig BGP refactor";
reference "2.1.1";
}

grouping bgp-common-neighbor-group-timers-config {
  description
  "Config parameters related to timers associated with the BGP peer";

  leaf connect-retry {
    type decimal64 {
      fraction-digits 2;
    }
    default 30;
    description
    "Time interval in seconds between attempts to establish a session with the peer.";
  }

  leaf hold-time {
    type decimal64 {
      fraction-digits 2;
    }
    default 90;
    description
    "Time interval in seconds that a BGP session will be considered active in the absence of keepalive or other messages from the peer. The hold-time is typically set to 3x the keepalive-interval.";
    reference
    "RFC 4271 - A Border Gateway Protocol 4, Sec. 10";
  }

  leaf keepalive-interval {
    type decimal64 {
      fraction-digits 2;
    }
    default 30;
    description
    "Time interval in seconds between transmission of keepalive messages to the neighbor. Typically set to 1/3 the hold-time.";
  }

  leaf minimum-advertisement-interval {
    type decimal64 {
      fraction-digits 2;
    }
  }
}
default 30;
description
"Minimum time which must elapse between subsequent UPDATE
messages relating to a common set of NLRI being transmitted
to a peer. This timer is referred to as
MinRouteAdvertisementIntervalTimer by RFC 4721 and serves to
reduce the number of UPDATE messages transmitted when a
particular set of NLRI exhibit instability.";
reference
"RFC 4271 - A Border Gateway Protocol 4, Sec 9.2.1.1";
}
}
grouping bgp-common-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 peer-type {
type oc-bgp-types:peer-type;
description
"Explicitly designate the peer or peer group as internal
(iBGP) or external (eBGP)."
}
leaf auth-password {
type string;
description
"Configures an MD5 authentication password for use with
neighboring devices."
}
leaf remove-private-as {
// could also make this a container with a flag to enable
// remove-private and separate option. here, option implies
// remove-private is enabled.
type oc-bgp-types:remove-private-as-option;
description
"Remove private AS numbers from updates sent to peers - when
this leaf is not specified, the AS_PATH attribute should be
sent to the peer unchanged";
}

leaf route-flap-damping {
  type boolean;
default false;
description
  "Enable route flap damping.";
}

leaf send-community {
  type oc-bgp-types:community-type;
default "NONE";
description
  "Specify which types of community should be sent to the
neighbor or group. The default is to not send the
community attribute";
}

leaf description {
  type string;
description
  "An optional textual description (intended primarily for use
with a peer or group";
}

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

  leaf tcp-mss {
    type uint16;
description
    "Sets the max segment size for BGP TCP sessions.";
  }

  leaf mtu-discovery {
    type boolean;
default false;
description
  "Turns path mtu discovery for BGP TCP sessions on (true) or off (false)";
}

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 local-address {
  type union {
    type inet:ip-address;
    type string;
  }
  //TODO: the string should be converted to a leafref type to point to an interface when YANG 1.1 is available with leafrefs in union types.
  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.";
}

grouping bgp-common-neighbor-group-error-handling-config {
  description
    "Configuration parameters relating to enhanced error handling behaviours for BGP";

  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 "draft-ietf-idr-error-handling-16";
  }
}

grouping bgp-common-neighbor-group-logging-options-config {
  description
    "Configuration parameters specifying the logging behaviour for
BGP sessions to the peer;

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."
}
}

grouping bgp-common-neighbor-group-multihop-config {
  description
  "Configuration parameters specifying the multihop behaviour for BGP sessions to the peer";

  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";
  }
}

grouping bgp-common-neighbor-group-route-reflector-config {
  description
  "Configuration parameters determining whether the behaviour of the local system when acting as a route-reflector";

  leaf route-reflector-cluster-id {
    type oc-bgp-types: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.";
  }

  leaf route-reflector-client {

type boolean;
default "false";
description
  "Configure the neighbor as a route reflector client.";
}

grouping bgp-common-neighbor-group-as-path-options-config {
  description
    "Configuration parameters allowing manipulation of the AS_PATH attribute";

  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.";
  }

  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";
  }
}

grouping bgp-common-neighbor-group-add-paths-config {
  description
    "Configuration parameters specifying whether the local system will send or receive multiple paths using ADD_PATHS";

  leaf receive {
    type boolean;
    default false;
    description
      "Enable ability to receive multiple path advertisements for an NLRI from the neighbor or group";
  }

  leaf send-max {
    type uint8;
    description
      "The maximum number of paths to advertise to neighbors for a single NLRI";
  }

leaf eligible-prefix-policy {
  type leafref {
    path "/rpol:routing-policy/rpol:policy-definitions/" + 
    "rpol:policy-definition/rpol:name";
  }
  description
  "A reference to a routing policy which can be used to
  restrict the prefixes for which add-paths is enabled";
}
}
grouping bgp-common-graceful-restart-config {
  description
  "Configuration parameters relating to BGP graceful restart.";

  leaf enabled {
    type boolean;
    description
    "Enable or disable the graceful-restart capability.";
  }

  leaf restart-time {
    type uint16 {
      range 0..4096;
    }
    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.";
  }

  leaf stale-routes-time {
    type decimal64 {
      fraction-digits 2;
    }
    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";
  }

  leaf helper-only {

type boolean;
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."
}
}
grouping bgp-common-use-multiple-paths-config {
description
"Generic configuration options relating to use of multiple paths for a referenced AFI-SAFI, group or neighbor";
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.";
}
}
grouping bgp-common-use-multiple-paths-ebgp-as-options-config {
description
"Configuration parameters specific to eBGP multipath applicable to all contexts";
leaf allow-multiple-as {
  type boolean;
default "false";
description
"Allow multipath to use paths from different neighbouring ASes. The default is to only consider multiple paths from the same neighbouring AS.";
}
}
grouping bgp-common-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;"

carrier container config {
  description
  "Configuration parameters relating to multipath";
  uses bgp-common-use-multiple-paths-config;
}
carrier container state {
  config false;
  description
  "State parameters relating to multipath";
  uses bgp-common-use-multiple-paths-config;
}

container ebgp {
  description
  "Multipath parameters for eBGP";
  container config {
    description
    "Configuration parameters relating to eBGP multipath";
    uses bgp-common-use-multiple-paths-ebgp-config;
  }
  container state {
    config false;
    description
    "State information relating to eBGP multipath";
    uses bgp-common-use-multiple-paths-ebgp-config;
  }
}

container ibgp {
  description
  "Multipath parameters for iBGP";
  container config {
    description
    "Configuration parameters relating to iBGP multipath";
    uses bgp-common-use-multiple-paths-ibgp-config;
  }
  container state {
    config false;
    description
    "State information relating to iBGP multipath";
    uses bgp-common-use-multiple-paths-ibgp-config;
  }
}
}
grouping bgp-common-use-multiple-paths-ebgp-config {
    description
    "Configuration parameters relating to multipath for eBGP";

    leaf allow-multiple-as {
        type boolean;
        default "false";
        description
        "Allow multipath to use paths from different neighbouring
         ASes. The default is to only consider multiple paths from
         the same neighbouring AS.";
    }

    leaf maximum-paths {
        type uint32;
        default 1;
        description
        "Maximum number of parallel paths to consider when using
         BGP multipath. The default is use a single path.";
    }
}

grouping bgp-common-use-multiple-paths-ibgp-config {
    description
    "Configuration parameters relating to multipath for iBGP";

    leaf maximum-paths {
        type uint32;
        default 1;
        description
        "Maximum number of parallel paths to consider when using
         iBGP multipath. The default is to use a single path";
    }
}

grouping bgp-common-route-selection-options-config {
    description
    "Set of configuration options that govern best
     path 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 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.";
}

leaf enable-aigp {
  type boolean;
  default false;
  description
    "Flag to enable sending / receiving accumulated IGP attribute in routing updates";
}

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";
}

...
"Configuration and state relating to route selection options";

container route-selection-options {
    description
    "Parameters relating to options for route selection";
    container config {
        description
        "Configuration parameters relating to route selection options";
        uses bgp-common-route-selection-options-config;
    }
    container state {
        config false;
        description
        "State information for the route selection options";
        uses bgp-common-route-selection-options-config;
    }
}

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

    leaf total-paths {
        type uint32;
        description
        "Total number of BGP paths within the context";
    }

    leaf total-prefixes {
        type uint32;
        description
        "Total number of BGP prefixes received within the context";
    }
}

<CODE ENDS>

<CODE BEGINS> file "ietf-bgp-common-multiprotocol.yang"

submodule ietf-bgp-common-multiprotocol {

belongs-to ietf-bgp {
  prefix "bgp";
}

import openconfig-extensions { prefix oc-ext; }
import ietf-bgp-types { prefix oc-bgp-types; }
import ietf-routing-policy { prefix rpol; }
import openconfig-types { prefix oc-types; }

include ietf-bgp-common;

// meta
organization
  "OpenConfig working group";

contact
  "OpenConfig working group
  netopenconfig@googlegroups.com";

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

oc-ext:openconfig-version "2.1.1";

revision "2016-06-21" {
  description
    "OpenConfig BGP refactor"
  reference "2.1.1";
}

grouping bgp-common-mp-afi-safi-graceful-restart-config {
  description
    "BGP graceful restart parameters that apply on a per-AFI-SAFI
    basis";

  leaf enabled {
    type boolean;
    default false;
    description
      "This leaf indicates whether graceful-restart is enabled for
      this AFI-SAFI";
  }
}

grouping bgp-common-mp-afi-safi-config {
  description
"Configuration parameters used for all BGP AFI-SAFIs"

leaf afi-safi-name {
  type identityref {
    base oc-bgp-types:AFI_SAFI_TYPE;
  }
  description "AFI,SAFI";
}

leaf enabled {
  type boolean;
  default false;
  description
    "This leaf indicates whether the IPv4 Unicast AFI,SAFI is
    enabled for the neighbour or group";
}

grouping bgp-common-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 rpol:apply-policy-group;

  uses bgp-common-mp-ipv4-unicast-group;
  uses bgp-common-mp-ipv6-unicast-group;
  uses bgp-common-mp-ipv4-labeled-unicast-group;
  uses bgp-common-mp-ipv6-labeled-unicast-group;
  uses bgp-common-mp-l3vpn-ipv4-unicast-group;
  uses bgp-common-mp-l3vpn-ipv6-unicast-group;
  uses bgp-common-mp-l3vpn-ipv4-multicast-group;
  uses bgp-common-mp-l3vpn-ipv6-multicast-group;
  uses bgp-common-mp-l2vpn-vpls-group;
  uses bgp-common-mp-l2vpn-evpn-group;
}

// Groupings relating to each address family

grouping bgp-common-mp-ipv4-unicast-group {
  description
    "Group for IPv4 Unicast configuration options";

  container ipv4-unicast {
    when "/afi-safi-name = 'oc-bgp-types:IPV4_UNICAST'" {
      description
        "Include this container for IPv4 Unicast specific
        configuration";
    }
  }
}
description "IPv4 unicast configuration options";

// include common IPv[46] unicast options
uses bgp-common-mp-ipv4-ipv6-unicast-common;

// placeholder for IPv4 unicast specific configuration
}
}

grouping bgp-common-mp-ipv6-unicast-group {
  description 
  "Group for IPv6 Unicast configuration options";

  container ipv6-unicast {
    when "../afi-safi-name = 'oc-bgp-types:IPV6_UNICAST’" 
      description 
      "Include this container for IPv6 Unicast specific configuration";
  }
}

description "IPv6 unicast configuration options";

// include common IPv[46] unicast options
uses bgp-common-mp-ipv4-ipv6-unicast-common;

// placeholder for IPv6 unicast specific configuration
// options
}
}

grouping bgp-common-mp-ipv4-labeled-unicast-group {
  description 
  "Group for IPv4 Labeled Unicast configuration options";

  container ipv4-labeled-unicast {
    when "../afi-safi-name = 'oc-bgp-types:IPV4_LABELED_UNICAST’" 
      description 
      "Include this container for IPv4 Labeled Unicast specific configuration";
  }
}

description "IPv4 Labeled Unicast configuration options";

uses bgp-common-mp-all-afi-safi-common;

// placeholder for IPv4 Labeled Unicast specific config
grouping bgp-common-mp-ipv6-labeled-unicast-group {
  description
  "Group for IPv6 Labeled Unicast configuration options";

  container ipv6-labeled-unicast {
    when ".../afi-safi-name = 'oc-bgp-types:IPV6_LABELED_UNICAST’" {
      description
      "Include this container for IPv6 Labeled Unicast specific
          configuration";
    }

    description "IPv6 Labeled Unicast configuration options";

    uses bgp-common-mp-all-afi-safi-common;

    // placeholder for IPv6 Labeled Unicast specific config
    // options.
  }
}

grouping bgp-common-mp-l3vpn-ipv4-unicast-group {
  description
  "Group for IPv4 Unicast L3VPN configuration options";

  container l3vpn-ipv4-unicast {
    when ".../afi-safi-name = 'oc-bgp-types: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 bgp-common-mp-l3vpn-ipv4-ipv6-unicast-common;

    // placeholder for IPv4 Unicast L3VPN specific config options.
  }
}

grouping bgp-common-mp-l3vpn-ipv6-unicast-group {
  description
  "Group for IPv6 Unicast L3VPN configuration options";
container l3vpn-ipv6-unicast {
  when "../afi-safi-name = 'oc-bgp-types: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 bgp-common-mp-l3vpn-ipv4-ipv6-unicast-common;

// placeholder for IPv6 Unicast L3VPN specific configuration
// options
}
}

grouping bgp-common-mp-l3vpn-ipv4-multicast-group {
  description
    "Group for IPv4 L3VPN multicast configuration options";

  container l3vpn-ipv4-multicast {
    when "../afi-safi-name = 'oc-bgp-types: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 bgp-common-mp-l3vpn-ipv4-ipv6-multicast-common;

// placeholder for IPv4 Multicast L3VPN specific configuration
// options
}
}

grouping bgp-common-mp-l3vpn-ipv6-multicast-group {
  description
    "Group for IPv6 L3VPN multicast configuration options";

  container l3vpn-ipv6-multicast {
    when "../afi-safi-name = 'oc-bgp-types: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 bgp-common-mp-l3vpn-ipv4-ipv6-multicast-common;

// placeholder for IPv6 Multicast L3VPN specific configuration
// options
}
}

grouping bgp-common-mp-l2vpn-vpls-group {
  description "Group for BGP-signalled VPLS configuration options";

  container l2vpn-vpls {
    when ".../afi-safi-name = 'oc-bgp-types:L2VPN_VPLS'" {
      description "Include this container for BGP-signalled VPLS specific configuration";
    }
  }

  description "BGP-signalled VPLS configuration options";

  // include common L2VPN options
  uses bgp-common-mp-l2vpn-common;

  // placeholder for BGP-signalled VPLS specific configuration
  // options
  }
}

grouping bgp-common-mp-l2vpn-evpn-group {
  description "Group for BGP EVPN configuration options";

  container l2vpn-evpn {
    when ".../afi-safi-name = 'oc-bgp-types:L2VPN_EVPN'" {
      description "Include this container for BGP EVPN specific configuration";
    }
  }

  description "BGP EVPN configuration options";

  // include common L2VPN options
  uses bgp-common-mp-l2vpn-common;

  // placeholder for BGP EVPN specific configuration options
  }
// Common groupings across multiple AFI,SAFIs

grouping bgp-common-mp-all-afi-safi-common {
    description
    "Grouping for configuration common to all AFI,SAFI";

container prefix-limit {
    description
    "Configure the maximum number of prefixes that will be accepted from a peer";

container config {
    description
    "Configuration parameters relating to the prefix limit for the AFI-SAIF";
    uses bgp-common-mp-all-afi-safi-common-prefix-limit-config;
}

container state {
    config false;
    description
    "State information relating to the prefix-limit for the AFI-SAIF";
    uses bgp-common-mp-all-afi-safi-common-prefix-limit-config;
}
}

grouping bgp-common-mp-ipv4-ipv6-unicast-common {
    description
    "Common configuration that is applicable for IPv4 and IPv6 unicast";

    // include common afi-safi options.
    uses bgp-common-mp-all-afi-safi-common;

    // configuration options that are specific to IPv[46] unicast
    container config {
        description
        "Configuration parameters for common IPv4 and IPv6 unicast AFI-SAIF options";
        uses bgp-common-mp-ipv4-ipv6-unicast-common-config;
    }
    container state {
        config false;
        description

"State information for common IPv4 and IPv6 unicast parameters";
uses bgp-common-mp-ipv4-ipv6-unicast-common-config;
}
}

grouping bgp-common-mp-l3vpn-ipv4-ipv6-unicast-common {
  description
  "Common configuration applied across L3VPN for IPv4 and IPv6";

  // placeholder -- specific configuration options that are generic
  uses bgp-common-mp-all-afi-safi-common;
}

grouping bgp-common-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 bgp-common-mp-all-afi-safi-common;
}

grouping bgp-common-mp-l2vpn-common {
  description
  "Common configuration applied across L2VPN address families";

  // placeholder -- specific configuration options that are generic
  // across L2VPN address families
  uses bgp-common-mp-all-afi-safi-common;
}

// Config groupings for common groups
grouping bgp-common-mp-all-afi-safi-common-prefix-limit-config {
  description
  "Configuration parameters relating to prefix-limits for an AFI-SAFI";

  leaf max-prefixes {
    type uint32;
    description
    "Maximum number of prefixes that will be accepted from the neighbour";
  }
}
leaf shutdown-threshold-pct {
    type oc-types:percentage;
    description
    "Threshold on number of prefixes that can be received
    from a neighbour before generation of warning messages
    or log entries. Expressed as a percentage of
    max-prefixes";
}

leaf restart-timer {
    type decimal64 {
        fraction-digits 2;
    }
    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 bgp-common-mp-ipv4-ipv6-unicast-common-config {
    description
    "Common configuration parameters for IPv4 and IPv6 Unicast
    address families";

    leaf send-default-route {
        type boolean;
        default "false";
        description
        "If set to true, send the default-route to the neighbour(s)";
    }
}

<CODE ENDS>

<CODE BEGINS> file "ietf-bgp-common-structure.yang"

submodule ietf-bgp-common-structure {

    belongs-to ietf-bgp {
        prefix "bgp";
    }

    import openconfig-extensions { prefix oc-ext; }

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

// meta
organization
  "OpenConfig working group";

contact
  "OpenConfig working group
   netopenconfig@googlegroups.com";

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

oc-ext:openconfig-version "2.1.1";

revision "2016-06-21" {
  description
    "OpenConfig BGP refactor";
  reference "2.1.1";
}

grouping bgp-common-structure-neighbor-group-logging-options {
  description
    "Structural grouping used to include error handling configuration and
     state for both BGP neighbors and groups";

container logging-options {
  description
    "Logging options for events related to the BGP neighbor or
     group";

container config {
  description
    "Configuration parameters enabling or modifying logging
     for events relating to the BGP group";
    uses bgp-common-neighbor-group-logging-options-config;
}

container state {
  config false;
  description
    "State information relating to logging for the BGP neighbor
     or group";
    uses bgp-common-neighbor-group-logging-options-config;
}
}
grouping bgp-common-structure-neighbor-group-ebgp-multihop {
    description "Structural grouping used to include eBGP multihop configuration and state for both BGP neighbors and peer groups";

    container ebgp-multihop {
        description "eBGP multi-hop parameters for the BGPGroup";
        container config {
            description "Configuration parameters relating to eBGP multihop for the BGP group";
            uses bgp-common-neighbor-group-multihop-config;
        }
        container state {
            config false;
            description "State information for eBGP multihop, for the BGP neighbor or group";
            uses bgp-common-neighbor-group-multihop-config;
        }
    }
}

grouping bgp-common-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 BGPGroup";
        container config {
            description "Configuration parameters relating to route reflection for the BGPGroup";
            uses bgp-common-neighbor-group-route-reflector-config;
        }
        container state {
            config false;
            description "State information relating to route reflection for the BGPGroup";
            uses bgp-common-neighbor-group-route-reflector-config;
        }
    }
}
grouping bgp-common-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";
    container config {
        description
        "Configuration parameters relating to AS_PATH manipulation for the BGP peer or group";
        uses bgp-common-neighbor-group-as-path-options-config;
    }
    container state {
        config false;
        description
        "State information relating to the AS_PATH manipulation mechanisms for the BGP peer or group";
        uses bgp-common-neighbor-group-as-path-options-config;
    }
}
}

grouping bgp-common-structure-neighbor-group-add-paths {
    description
    "Structural grouping used to include ADD-PATHs configuration and state for both BGP neighbors and peer groups";

container add-paths {
    description
    "Parameters relating to the advertisement and receipt of multiple paths for a single NLRI (add-paths)";
    container config {
        description
        "Configuration parameters relating to ADD_PATHS";
        uses bgp-common-neighbor-group-add-paths-config;
    }
    container state {
        config false;
        description
        "State information associated with ADD_PATHS";
        uses bgp-common-neighbor-group-add-paths-config;
    }
}
}
submodule ietf-bgp-peer-group {

    belongs-to ietf-bgp {
        prefix "bgp";
    }

    import openconfig-extensions { prefix oc-ext; }
    import ietf-routing-policy { prefix rpol; }

    // Include the common submodule
    include ietf-bgp-common;
    include ietf-bgp-common-multiprotocol;
    include ietf-bgp-common-structure;

    // meta
    organization
        "OpenConfig working group";

    contact
        "OpenConfig working group
        netopenconfig@googlegroups.com";

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

    oc-ext:openconfig-version "2.1.1";

    revision "2016-06-21" {
        description
            "OpenConfig BGP refactor";
        reference "2.1.1";
    }

    grouping bgp-peer-group-config {
        description
            "Configuration parameters relating to a base BGP peer group that
            are not also applicable to any other context (e.g., neighbor)";

        leaf peer-group-name {
            type string;
        }
    }

    // Include the common submodule
    include ietf-bgp-common;
    include ietf-bgp-common-multiprotocol;
    include ietf-bgp-common-structure;

    // meta
    organization
        "OpenConfig working group";

    contact
        "OpenConfig working group
        netopenconfig@googlegroups.com";

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

    oc-ext:openconfig-version "2.1.1";

    revision "2016-06-21" {
        description
            "OpenConfig BGP refactor";
        reference "2.1.1";
    }

    grouping bgp-peer-group-config {
        description
            "Configuration parameters relating to a base BGP peer group that
            are not also applicable to any other context (e.g., neighbor)";

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

}

grouping bgp-peer-group-afi-safi-list { description
  "List of address-families associated with the BGP peer-group";
}

list afi-safi { key "afi-safi-name";

description
  "AFI,SAFI configuration available for the neighbour or group";

leaf afi-safi-name {
  type leafref {
    path ".../config/afi-safi-name";
  }
  description
    "Reference to the AFI-SAFI name used as a key for the AFI-SAFI list";
}

container config {
  description
    "Configuration parameters for the AFI-SAFI";
  uses bgp-common-mp-afi-safi-config;
}

container state {
  config false;
  description
    "State information relating to the AFI-SAFI";
  uses bgp-common-mp-afi-safi-config;
}

container graceful-restart {
  description
    "Parameters relating to BGP graceful-restart";
  container config {
    description
      "Configuration options for BGP graceful-restart";
    uses bgp-common-mp-afi-safi-graceful-restart-config;
  }
  container state {
    config false;
description
  "State information for BGP graceful-restart";
  uses bgp-common-mp-afi-safi-graceful-restart-config;
}
}

uses bgp-common-route-selection-options;
uses bgp-common-global-group-use-multiple-paths;
uses bgp-common-mp-all-afi-safi-list-contents;
}
}

grouping bgp-peer-group-base {
  description
   "Parameters related to a BGP group";

  container config {
    description
      "Configuration parameters relating to the BGP neighbor or
group";
    uses bgp-peer-group-config;
    uses bgp-common-neighbor-group-config;
  }

  container state {
    config false;
    description
      "State information relating to the BGP peer-group";
    uses bgp-peer-group-config;
    uses bgp-common-neighbor-group-config;
    uses bgp-common-state;
  }

  container timers {
    description
      "Timers related to a BGP peer-group";

    container config {
      description
        "Configuration parameters relating to timers used for the
BGP neighbor or peer group";
      uses bgp-common-neighbor-group-timers-config;
    }

    container state {
      config false;
      description
        "State information relating to the timers used for the BGP
group";
      uses bgp-common-neighbor-group-timers-config;
    }
}
container transport {
    description
    "Transport session parameters for the BGP peer-group";

    container config {
        description
        "Configuration parameters relating to the transport
        session(s) used for the BGP neighbor or group";
        uses bgp-common-neighbor-group-transport-config;
    }

    container state {
        config false;
        description
        "State information relating to the transport session(s)
        used for the BGP neighbor or group";
        uses bgp-common-neighbor-group-transport-config;
    }
}

container error-handling {
    description
    "Error handling parameters used for the BGP peer-group";

    container config {
        description
        "Configuration parameters enabling or modifying the
        behavior or enhanced error handling mechanisms for the BGP
        group";
        uses bgp-common-neighbor-group-error-handling-config;
    }

    container state {
        config false;
        description
        "State information relating to enhanced error handling
        mechanisms for the BGP group";
        uses bgp-common-neighbor-group-error-handling-config;
    }
}

container graceful-restart {
    description
    "Parameters relating the graceful restart mechanism for BGP";

    container config {
        description
        "Configuration parameters relating to graceful-restart";
    }
}
uses bgp-common-graceful-restart-config;
}
container state {
  config false;
  description
    "State information associated with graceful-restart";
  uses bgp-common-graceful-restart-config;
}
}

uses bgp-common-structure-neighbor-group-logging-options;
uses bgp-common-structure-neighbor-group-ebgp-multihop;
uses bgp-common-structure-neighbor-group-route-reflector;
uses bgp-common-structure-neighbor-group-as-path-options;
uses bgp-common-structure-neighbor-group-add-paths;
uses bgp-common-global-group-use-multiple-paths;
uses rpol:apply-policy-group;

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

grouping bgp-peer-group-list {
  description
    "The list of BGP peer groups";

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

leaf peer-group-name {
  type leafref {
    path "../config/peer-group-name";
  }
  description
    "Reference to the name of the BGP peer-group used as a
    key in the peer-group list";
  }
}

uses bgp-peer-group-base;
submodule ietf-bgp-neighbor {

  belongs-to ietf-bgp {
    prefix "bgp";
  }

  import openconfig-extensions { prefix oc-ext; }
  import ietf-routing-policy { prefix rpol; }
  import openconfig-types { prefix oc-types; }
  import ietf-bgp-types { prefix oc-bgp-types; }
  import ietf-inet-types { prefix inet; }
  import ietf-yang-types { prefix yang; }

  // Include the common submodule
  include ietf-bgp-common;
  include ietf-bgp-common-multiprotocol;
  include ietf-bgp-peer-group;
  include ietf-bgp-common-structure;

  // meta
  organization
    "OpenConfig working group";

  contact
    "OpenConfig working group
     netopenconfig@googlegroups.com";

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

  oc-ext:openconfig-version "2.1.1";

  revision "2016-06-21" { 
    description
      "OpenConfig BGP refactor";
    reference "2.1.1";
  }

  grouping bgp-neighbor-config {
    description

"Configuration parameters relating to a base BGP neighbor that are not also applicable to any other context (e.g., peer group)");

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

leaf neighbor-address {
  type inet:ip-address;
  description
  "Address of the BGP peer, either in IPv4 or IPv6";
}

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.";
}

grouping bgp-neighbor-use-multiple-paths {
  description
  "Multipath 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";

container config {
  description
  "Configuration parameters relating to multipath";
  uses bgp-common-use-multiple-paths-config;
}

container state {

container ebgp {
    description
    "Multipath configuration for eBGP";
    container config {
        description
        "Configuration parameters relating to eBGP multipath";
        uses bgp-common-use-multiple-paths-ebgp-as-options-config;
    }
    container state {
        config false;
        description
        "State information relating to eBGP multipath";
        uses bgp-common-use-multiple-paths-ebgp-as-options-config;
    }
}

grouping bgp-neighbor-state {
    description
    "Operational state parameters relating only to a BGP neighbor";

    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";
}
}
description
"Operational state of the BGP peer";
}
leaf last-established {
    type oc-types:timeticks64;
    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).";
}
leaf established-transitions {
    type yang:counter64;
    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";
}
leaf-list supported-capabilities {
    type identityref {
        base oc-bgp-types:BGP_CAPABILITY;
    }
    description

"BGP capabilities negotiated as supported with the peer";
}

container messages {
description "Counters for BGP messages sent and received from the neighbor";
carriertosent {
description "Counters relating to BGP messages sent to the neighbor";
uses bgp-neighbor-counters-message-types-state;
}

carriertoreceived {
description "Counters for BGP messages received from the neighbor";
uses bgp-neighbor-counters-message-types-state;
}
}

container queues {
description "Counters related to queued messages associated with the BGP neighbor";
uses bgp-neighbor-queue-counters-state;
}


grouping bgp-neighbor-counters-message-types-state {
description "Grouping of BGP message types, included for re-use across counters";
leaf UPDATE {
type uint64;
description "Number of BGP UPDATE messages announcing, withdrawing or modifying paths exchanged.";
}

leaf NOTIFICATION {
type uint64;
description "Number of BGP NOTIFICATION messages indicating an error condition has occurred exchanged.";
}
}
grouping bgp-neighbor-queue-counters-state {
  description
    "Counters relating to the message queues associated with the BGP peer";

  leaf input {
    type uint32;
    description
      "The number of messages received from the peer currently queued";
  }

  leaf output {
    type uint32;
    description
      "The number of messages queued to be sent to the peer";
  }
}


grouping bgp-neighbor-transport-state {
  description
    "Operational state parameters relating to the transport session used for the BGP session";

  leaf local-port {
    type inet:port-number;
    description
      "Local TCP port being used for the TCP session supporting the BGP session";
  }

  leaf remote-address {
    type inet:ip-address;
    description
      "Remote address to which the BGP session has been established";
  }

  leaf remote-port {
    type inet:port-number;
    description
      "Remote port being used by the peer for the TCP session supporting the BGP session";
  }
}


grouping bgp-neighbor-error-handling-state {
  description

"Operational state parameters relating to enhanced error error handling for BGP";

leaf erroneous-update-messages {
  type uint32;
  description
  "The number of BGP UPDATE messages for which the treat-as-withdraw mechanism has been applied based on erroneous message contents";
}
}

grouping bgp-neighbor-timers-state {
  description
  "Operational state parameters relating to BGP timers associated with the BGP session";

  leaf negotiated-hold-time {
    type decimal64 { fraction-digits 2; }
    description
    "The negotiated hold-time for the BGP session";
  }
}


grouping bgp-neighbor-afi-safi-graceful-restart-state {
  description
  "Operational state variables relating to the graceful-restart mechanism on a per-AFI-SAFI basis";

  leaf received {
    type boolean;
    description
    "This leaf indicates whether the neighbor advertised the ability to support graceful-restart for this AFI-SAFI";
  }

  leaf advertised {
    type boolean;
    description
    "This leaf indicates whether the ability to support graceful-restart has been advertised to the peer";
  }
}


grouping bgp-neighbor-graceful-restart-state {
  description

"Operational state information relevant to graceful restart for BGP";

leaf peer-restart-time {
    type uint16 {
        range 0..4096;
    }
    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;
    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;
    description
    "This flag indicates whether the local neighbor is currently restarting. The flag is unset after all NLRI have been advertised to the peer, and the End-of-RIB (EOR) marker has been unset";
}

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 router is operating in remote helper mode, and hence will retain forwarding state during a remote restart, but will not retain forwarding state during a local session restart";
        }
    }
}

"The local system is able to retain routes during restart but the remote system is only able to act as a helper";

grouping bgp-neighbor-afi-safi-state {
  description
  "Operational state parameters relating to an individual AFI, SAFI for a neighbor";

  leaf active {
    type boolean;
    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 {
    description "Prefix counters for the BGP session";
    leaf received {
      type uint32;
      description
      "The number of prefixes received from the neighbor";
    }

    leaf sent {
      type uint32;
      description
      "The number of prefixes advertised to the neighbor";
    }

    leaf installed {
      type uint32;
      description
      "The number of advertised prefixes installed in the Loc-rib";
    }
  }
}
grouping bgp-neighbor-afi-safi-list {
  description
    "List of address-families associated with the BGP neighbor";

  list afi-safi {
    key "afi-safi-name";

    description
      "AFI,SAFI configuration available for the neighbour or group";

    leaf afi-safi-name {
      type leafref {
        path "../config/afi-safi-name";
      }

      description
        "Reference to the AFI-SAFI name used as a key for the AFI-SAFI list";

    }

    container config {
      description
        "Configuration parameters for the AFI-SAFI";

      uses bgp-common-mp-afi-safi-config;
    }

    container state {
      config false;

      description
        "State information relating to the AFI-SAFI";

      uses bgp-common-mp-afi-safi-config;

      uses bgp-neighbor-afi-safi-state;
    }

    container graceful-restart {
      description
        "Parameters relating to BGP graceful-restart";

      container config {
        description
          "Configuration options for BGP graceful-restart";

        uses bgp-common-mp-afi-safi-graceful-restart-config;
      }

      container state {
        config false;

        description
          "State information for BGP graceful-restart";

        uses bgp-common-mp-afi-safi-graceful-restart-config;
      }
    }
  }
}
uses bgp-neighbor-afi-safi-graceful-restart-state;
}
}

uses bgp-common-mp-all-afi-safi-list-contents;
uses bgp-neighbor-use-multiple-paths;
}
}

grouping bgp-neighbor-base {
  description
  "Parameters related to a BGP neighbor";
  container config {
    description
    "Configuration parameters relating to the BGP neighbor or group";
    uses bgp-neighbor-config;
    uses bgp-common-neighbor-group-config;
  }
  container state {
    config false;
    description
    "State information relating to the BGP neighbor";
    uses bgp-neighbor-config;
    uses bgp-common-neighbor-group-config;
    uses bgp-neighbor-state;
  }
}

container timers {
  description
  "Timers related to a BGP neighbor";
  container config {
    description
    "Configuration parameters relating to timers used for the BGP neighbor";
    uses bgp-common-neighbor-group-timers-config;
  }
  container state {
    config false;
    description
    "State information relating to the timers used for the BGP neighbor";
    uses bgp-common-neighbor-group-timers-config;
    uses bgp-neighbor-timers-state;
  }
}
container transport {
  description
  "Transport session parameters for the BGP neighbor";
  container config {
    description
    "Configuration parameters relating to the transport session(s) used for the BGP neighbor";
    uses bgp-common-neighbor-group-transport-config;
  }
  container state {
    config false;
    description
    "State information relating to the transport session(s) used for the BGP neighbor";
    uses bgp-common-neighbor-group-transport-config;
    uses bgp-neighbor-transport-state;
  }
}

container error-handling {
  description
  "Error handling parameters used for the BGP neighbor or group";
  container config {
    description
    "Configuration parameters enabling or modifying the behavior or enhanced error handling mechanisms for the BGP neighbor";
    uses bgp-common-neighbor-group-error-handling-config;
  }
  container state {
    config false;
    description
    "State information relating to enhanced error handling mechanisms for the BGP neighbor";
    uses bgp-common-neighbor-group-error-handling-config;
    uses bgp-neighbor-error-handling-state;
  }
}

container graceful-restart {
  description
  "Parameters relating the graceful restart mechanism for BGP";
  container config {
    description
    "Configuration parameters relating to graceful-restart";
    uses bgp-common-graceful-restart-config;
  }
}
container state {
  config false;
  description "State information associated with graceful-restart";
  uses bgp-common-graceful-restart-config;
  uses bgp-neighbor-graceful-restart-state;
}

uses bgp-common-structure-neighbor-group-logging-options;
uses bgp-common-structure-neighbor-group-ebgp-multihop;
uses bgp-common-structure-neighbor-group-route-reflector;
uses bgp-common-structure-neighbor-group-as-path-options;
uses bgp-common-structure-neighbor-group-add-paths;
uses bgp-neighbor-use-multiple-paths;
uses rpol:apply-policy-group;

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

grouping bgp-neighbor-list {
  description "The list of BGP neighbors";

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

    leaf neighbor-address {
      type leafref {
        path "../config/neighbor-address";
      }
      description "Reference to the address of the BGP neighbor used as a key in the neighbor list";
    }

    uses bgp-neighbor-base;
  }
}
<CODE BEGINS> file "ietf-bgp-global.yang"

submodule ietf-bgp-global {

  belongs-to ietf-bgp {
    prefix "bgp";
  }

  import openconfig-extensions { prefix oc-ext; }
  import ietf-inet-types { prefix inet; }
  import ietf-yang-types { prefix yang; }

  // Include common submodule
  include ietf-bgp-common;
  include ietf-bgp-common-multiprotocol;

  // meta
  organization
    "OpenConfig working group";

  contact
    "OpenConfig working group
      netopenconfig@googlegroups.com";

  description
    "This sub-module contains groupings that are specific to the
     global context of the OpenConfig BGP module";

  oc-ext:openconfig-version "2.1.1";

  revision "2016-06-21" {
    description
      "OpenConfig BGP refactor";
    reference "2.1.1";
  }

  grouping bgp-global-config {
    description
      "Global configuration options for the BGP router.";

    leaf as {
      type inet:as-number;
      mandatory true;
    }

<CODE ENDS>
description
"Local autonomous system number of the router. Uses the 32-bit as-number type from the model in RFC 6991.";
}

leaf router-id {
  type yang:dotted-quad;
  description
  "Router id of the router - an unsigned 32-bit integer expressed in dotted quad notation.";
  reference
  "RFC4271 - A Border Gateway Protocol 4 (BGP-4), Section 4.2";
}

grouping bgp-global-state {
  description
  "Operational state parameters for the BGP neighbor";

  uses bgp-common-state;
}

grouping bgp-global-default-route-distance-config {
  description
  "Configuration options relating to the administrative distance (or preference) assigned to routes received from different sources (external, internal, and local).";

  leaf external-route-distance {
    type uint8 {
      range "1..255";
    }
    description
    "Administrative distance for routes learned from external BGP (eBGP).";
  }

  leaf internal-route-distance {
    type uint8 {
      range "1..255";
    }
    description
    "Administrative distance for routes learned from internal BGP (iBGP).";
  }
}

  grouping bgp-global-confederation-config {
"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."
}


grouping bgp-global-afi-safi-list {
  description
  "List of address-families associated with the BGP instance."

  list afi-safi {
    key "afi-safi-name";
    description
    "AFI,SAFI configuration available for the neighbour or group."

    leaf afi-safi-name {
      type leafref {
        path "../config/afi-safi-name";
      }
      description
      "Reference to the AFI-SAFI name used as a key for the AFI-SAFI list."
    }

    container config {
      description
      
    }
  }
}
"Configuration parameters for the AFI-SAFI";
uses bgp-common-mp-afi-safi-config;
}

container state {
    config false;
    description
        "State information relating to the AFI-SAFI";
    uses bgp-common-mp-afi-safi-config;
    uses bgp-common-state;
}

container graceful-restart {
    description
        "Parameters relating to BGP graceful-restart";
    container config {
        description
            "Configuration options for BGP graceful-restart";
        uses bgp-common-mp-afi-safi-graceful-restart-config;
    }
    container state {
        config false;
        description
            "State information for BGP graceful-restart";
        uses bgp-common-mp-afi-safi-graceful-restart-config;
    }
}

uses bgp-common-route-selection-options;
uses bgp-common-global-group-use-multiple-paths;
uses bgp-common-mp-all-afi-safi-list-contents;
}
}

// Structural groupings

grouping bgp-global-base {
    description
        "Global configuration parameters for the BGP router";

container config {
    description
        "Configuration parameters relating to the global BGP router";
    uses bgp-global-config;
}

container state {
    config false;
    description
        "State information relating to the global BGP router";
    uses bgp-global-config;
}
uses bgp-global-state;
)

container default-route-distance {
    description
    "Administrative distance (or preference) assigned to
    routes received from different sources
    (external, internal, and local).";

    container config {
        description
        "Configuration parameters relating to the default route
distance";
        uses bgp-global-default-route-distance-config;
    }

    container state {
        config false;
        description
        "State information relating to the default route distance";
        uses bgp-global-default-route-distance-config;
    }
}

container confederation {
    description
    "Parameters indicating whether the local system acts as part
    of a BGP confederation";

    container config {
        description
        "Configuration parameters relating to BGP confederations";
        uses bgp-global-confederation-config;
    }

    container state {
        config false;
        description
        "State information relating to the BGP confederations";
        uses bgp-global-confederation-config;
    }
}

container graceful-restart {
    description
    "Parameters relating the graceful restart mechanism for BGP";

    container config {
        description
        "Configuration parameters relating to graceful-restart";
        uses bgp-common-graceful-restart-config;
    }

    container state {
        config false;
        description
        "State information relating to graceful-restart";
        uses bgp-common-graceful-restart-config;
    }
}

8. BGP types

<CODE BEGINS> file "ietf-bgp-types.yang"

module ietf-bgp-types {
    yang-version "1";

    namespace "urn:ietf:params:xml:ns:yang:ietf-bgp-types";

    prefix "oc-bgp-types";

    import ietf-inet-types { prefix inet; }
    import openconfig-types { prefix oc-types; }
    import openconfig-extensions { prefix oc-ext; }

    // meta
    organization
        "OpenConfig working group";

    contact
        "OpenConfig working group
        netopenconfig@googlegroups.com";

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

oc-ext:openconfig-version "2.1.1"

revision "2016-06-21" {
    description
        "OpenConfig BGP refactor"
        reference "2.1.1"
}

identity BGP_CAPABILITY {
    description "Base identity for a BGP capability"
}

identity MPBGP {
    base BGP_CAPABILITY;
    description
        "Multi-protocol extensions to BGP"
        reference "RFC2858"
}

identity ROUTE_REFRESH {
    base BGP_CAPABILITY;
    description
        "The BGP route-refresh functionality"
        reference "RFC2918"
}

identity ASN32 {
    base BGP_CAPABILITY;
    description
        "4-byte (32-bit) AS number functionality"
        reference "RFC6793"
}

identity GRACEFUL_RESTART {
    base BGP_CAPABILITY;
    description
        "Graceful restart functionality"
        reference "RFC4724"
}

identity ADD_PATHS {
    base BGP_CAPABILITY;
    description
        "BGP add-paths";}
identity AFI_SAFI_TYPE {
    description "Base identity type for AFI,SAFI tuples for BGP-4";
    reference "RFC4760 - multiprotocol extensions for BGP-4";
}

identity IPV4_UNICAST {
    base AFI_SAFI_TYPE;
    description "IPv4 unicast (AFI,SAFI = 1,1)";
    reference "RFC4760";
}

identity IPV6_UNICAST {
    base AFI_SAFI_TYPE;
    description "IPv6 unicast (AFI,SAFI = 2,1)";
    reference "RFC4760";
}

identity IPV4_LABELED_UNICAST {
    base AFI_SAFI_TYPE;
    description "Labeled IPv4 unicast (AFI,SAFI = 1,4)";
    reference "RFC3107";
}

identity IPV6_LABELED_UNICAST {
    base AFI_SAFI_TYPE;
    description "Labeled IPv6 unicast (AFI,SAFI = 2,4)";
    reference "RFC3107";
}

identity L3VPN_IPV4_UNICAST {
    base AFI_SAFI_TYPE;
    description "Unicast IPv4 MPLS L3VPN (AFI,SAFI = 1,128)";
    reference "RFC4364";
}

identity L3VPN_IPV6_UNICAST {
    base AFI_SAFI_TYPE;
    description "Unicast IPv6 MPLS L3VPN (AFI,SAFI = 2,128)";
reference "RFC4659";
}

identity L3VPN_IPV4_MULTICAST {
    base AFI_SAFI_TYPE;
    description
        "Multicast IPv4 MPLS L3VPN (AFI,SAFI = 1,129)";
    reference "RFC6514";
}

identity L3VPN_IPV6_MULTICAST {
    base AFI_SAFI_TYPE;
    description
        "Multicast IPv6 MPLS L3VPN (AFI,SAFI = 2,129)";
    reference "RFC6514";
}

identity L2VPN_VPLS {
    base AFI_SAFI_TYPE;
    description
        "BGP-signalled VPLS (AFI,SAFI = 25,65)";
    reference "RFC4761";
}

identity L2VPN_EVPN {
    base AFI_SAFI_TYPE;
    description
        "BGP MPLS Based Ethernet VPN (AFI,SAFI = 25,70)";
}

identity BGP_WELL_KNOWN_STD_COMMUNITY {
    description
        "Reserved communities within the standard community space
         defined by RFC1997. These communities must fall within the
         range 0x00000000 to 0xFFFFFFFF";
    reference "RFC1997";
}

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 0xFFFFFFFF01.";
    reference "RFC1997";
}
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
     0xFFFFFFFF02.";
  reference "RFC1997";
}

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 0xFFFFFFFF03.";
  reference "RFC1997";
}

identity NOPEER {
  base BGP_WELL_KNOWN_STD_COMMUNITY;
  description
    "An autonomous system receiving NLRI tagged with this community
     is advised not to readvertise the NLRI to external bi-lateral
     peer autonomous systems. An AS may also filter received NLRI
     from bilateral peer sessions when they are tagged with this
     community value";
  reference "RFC3765";
}

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

typedef bgp-well-known-community-type {
  type identityref {
    base BGP_WELL_KNOWN_STD_COMMUNITY;
  }
}
typedef bgp-std-community-type {
// TODO: further refine restrictions and allowed patterns
// 4-octet value:
//  <as number> 2 octets
//  <community value> 2 octets
    type union {
        type uint32 {
            // per RFC 1997, 0x00000000 - 0x0000FFFF and 0xFFFF0000 -
            // 0xFFFFFFFF are reserved
            range "65536..4294901759"; // 0x00010000..0xFFFFEFFF
        }
        type string {
            pattern "([0-9]+:[0-9]+)";
        }
    }

description
"Type definition for standard commmunity attributes";
reference "RFC 1997 - BGP Communities Attribute";
}

typedef bgp-ext-community-type {
// TODO: needs more work to make this more precise given the
// variability of extended community attribute specifications
// 8-octet value:
//  <type> 2 octects
//  <value> 6 octets
    type union {
        type string {
            // Type 1: 2-octet global and 4-octet local
            // (AS number) (Integer)
            pattern "(6[0-5][0-5][0-3][0-5][0-5][0-5][0-9][0-9])' +
            '([1-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9])' +
            '([1-3][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9])' +
            '([1-3][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9])' +
        }
        type string {
            // Type 2: 4-octet global and 2-octet local
            // (ipv4-address) (integer)
            pattern "(([0-9][1-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9])' +
        }
    }

description
"Type definition for well-known IETF community attribute values";
reference "IANA Border Gateway Protocol (BGP) Well Known Communities";
}
typedef bgp-community-regexp-type {
    // TODO: needs more work to decide what format these regexps can
    // take.
}

type string {
    // route-target with Type 1
    // route-target:(ASN):(local-part)
    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]|1-9)';
}

type string {
    // route-target with Type 2
    // route-target:(IPv4):(local-part)
    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]|1-9)';
}

type string {
    // route-origin with Type 1
    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]|1-9)';
}

type string {
    // route-origin with Type 2
    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]|1-9)';
}

description
"Type definition for extended community attributes";
reference "RFC 4360 - BGP Extended Communities Attribute";
typedef oc-types:std-regexp;

description
"Type definition for communities specified as regular expression patterns";
}

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";
}

typedef peer-type {
  type enumeration {
    enum INTERNAL {
      description "internal (iBGP) peer";
    }
    enum EXTERNAL {
      description "external (eBGP) peer";
    }
  }
} 

description
"labels a peer or peer group as explicitly internal or external";
}

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."
}

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";
}

typedef percentage {
    type uint8 {
        range "0..100";
    }
    description
        "Integer indicating a percentage value";
}

typedef rr-cluster-id-type {
    type union {
        type uint32;
        type inet:ipv4-address;
    }
    description
        "union type for route reflector cluster ids:
         option 1: 4-byte number
         option 2: IP address";
}

typedef community-type {
    type enumeration {
        enum STANDARD {
            description "send only standard communities";
        }
    }
}
enum EXTENDED {
    description "send only extended communities";
}
enum BOTH {
    description "send both standard and extended communities";
}
enum NONE {
    description "do not send any community attribute";
}

description
"type describing variations of community attributes:
STANDARD: standard BGP community [rfc1997]
EXTENDED: extended BGP community [rfc4360]
BOTH: both standard and extended community";
}

<CODE ENDS>

9. BGP policy data

<CODE BEGINS> file "ietf-bgp-policy.yang"

module ietf-bgp-policy {

    yang-version "1";

    // namespace

    prefix "oc-bgp-pol";

    // import some basic types
    import ietf-inet-types { prefix inet; }
    import ietf-routing-policy {prefix rpol; }
    import openconfig-policy-types { prefix pol-types; }
    import ietf-bgp-types { prefix oc-bgp-types; }
    import openconfig-extensions { prefix oc-ext; }

    // meta
    organization
        "OpenConfig working group";

    contact
        "OpenConfig working group";

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

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.";
    }
  }
}
typedef bgp-next-hop-type {
  type union {
    type inet:ip-address-no-zone;
    type enumeration {
      enum SELF {
      }
    }
  }
}
typedef bgp-set-med-type {
    type union {
        type uint32;
        type string {
            pattern "^[+-][0-9]+";
        }
        type enumeration {
            enum IGP {
                description "set the MED value to the IGP cost toward the next hop for the route";
            }
        }
    }
}

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 statements

grouping match-community-config {
    description "Configuration data for match conditions on communities";
    leaf community-set {
        type leafref {
        description "References a defined community set";
    }

    uses rpol:match-set-options-group;
}
grouping match-community-state {
  description
    "Operational state data for match conditions on communities";
}

grouping match-community-top {
  description
    "Top-level grouping for match conditions on communities";

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";

  container config {
    description
      "Configuration data for match conditions on communities";

    uses match-community-config;
  }
}

container state {
  config false;

  description
    "Operational state data ";

  uses match-community-config;
  uses match-community-state;
}
}

grouping match-ext-community-config {
  description
    "Configuration data for match conditions on extended communities";

  leaf ext-community-set {
    type leafref {
      path "/rpol:routing-policy/rpol:defined-sets/" +
        "oc-bgp-pol:bgp-defined-sets/oc-bgp-pol:ext-community-sets/" +
        "oc-bgp-pol:ext-community-set/" +
        "oc-bgp-pol:ext-community-set-name";
    }
    description "References a defined extended community set";
  }
}
uses rpol:match-set-options-group;
}

grouping match-ext-community-state {
  description
  "Operational state data for match conditions on extended communities";
}

grouping match-ext-community-top {
  description
  "Top-level grouping for match conditions on extended communities";
}

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

  container config {
    description
    "Configuration data for match conditions on extended communities";

    uses match-ext-community-config;
  }

  container state {
    config false;
    description
    "Operational state data for match conditions on extended communities";

    uses match-ext-community-config;
    uses match-ext-community-state;
  }
}

grouping match-as-path-config {
  description
  "Configuration data for match conditions on AS path set";

  leaf as-path-set {

type leafref {
    path "/rpol:routing-policy/rpol:defined-sets/" +
    "oc-bgp-pol:bgp-defined-sets/oc-bgp-pol:as-path-sets/" +
    "oc-bgp-pol:as-path-set/oc-bgp-pol:as-path-set-name";
}
description "References a defined AS path set";
uses rpol:match-set-options-group;
}

grouping match-as-path-state {
    description
        "Operational state data for match conditions on AS path set";
}

grouping match-as-path-top {
    description
        "Top-level grouping for match conditions on AS path set";

container match-as-path-set {
    description
        "Match a referenced as-path set according to the logic defined in the match-set-options leaf";

container config {
    description
        "Configuration data for match conditions on AS path set";

    uses match-as-path-config;
}
}

container state {
    config false;

    description
        "Operational state data for match conditions on AS path set";

    uses match-as-path-config;
    uses match-as-path-state;
}
}

grouping bgp-match-set-conditions {
    description
        "Condition statement definitions for checking membership in a
defined set;

uses match-community-top;
uses match-ext-community-top;
uses match-as-path-top;
}

grouping community-count-config {
  description
    "Configuration data for community count condition";

  uses pol-types:attribute-compare-operators;
}

grouping community-count-state {
  description
    "Operational state data for community count condition";
}

grouping community-count-top {
  description
    "Top-level grouping for community count condition";

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

    container config {
      description
        "Configuration data for community count condition";

      uses community-count-config;
    }

    container state {
      config false;

      description
        "Operational state data for community count condition";

      uses community-count-config;
      uses community-count-state;
    }
  }
}
grouping as-path-length-config {
    description
    "Configuration data for AS path length condition";
    uses pol-types:attribute-compare-operators;
}

grouping as-path-length-state {
    description
    "Operational state data for AS path length condition";
}

grouping as-path-length-top {
    description
    "Top-level grouping for AS path length condition";

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

        container config {
            description
            "Configuration data for AS path length condition";

            uses as-path-length-config;
        }

        container state {
            config false;
            description
            "Operational state data for AS path length condition";

            uses as-path-length-config;
            uses as-path-length-state;
        }
    }
}

grouping bgp-conditions-config {
    description
    "Configuration data for BGP-specific policy conditions";

    leaf med-eq {
        type uint32;
        description
        "Med-eq leaf data for BGP-specific policy conditions";
    }

    leaf med-neq {
        type uint32;
        description
        "Med-neq leaf data for BGP-specific policy conditions";
    }
}

leaf med-eq {
    type uint32;
    description
    "Med-eq leaf data for BGP-specific policy conditions";
}

leaf med-neq {
    type uint32;
    description
    "Med-neq leaf data for BGP-specific policy conditions";
}
"Condition to check if the received MED value is equal to the specified value";
}

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

leaf-list next-hop-in {
  type inet:ip-address-no-zone;
  description
    "List of next hop addresses to check for in the route update";
}

leaf-list afi-safi-in {
  type identityref {
    base oc-bgp-types:AFI_SAFI_TYPE;
  }
  description
    "List of address families which the NLRI may be within";
}

leaf local-pref-eq {
  type uint32;
  // TODO: add support for other comparisons if needed
  description
    "Condition to check if the local pref attribute is equal to the specified value";
}

leaf route-type {
  // TODO: verify extent of vendor support for this comparison
  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";
}
grouping bgp-conditions-state {
  description
    "Operational state data for BGP-specific policy conditions";
}

grouping bgp-conditions-top {
  description
    "Top-level grouping for BGP-specific policy conditions";

custom bgp-conditions {
  description
    "Top-level container ";

custom config {
  description
    "Configuration data for BGP-specific policy conditions";

    uses bgp-conditions-config;
  }

custom state {
  config false;

  description
    "Operational state data for BGP-specific policy conditions";

    uses bgp-conditions-config;
    uses bgp-conditions-state;
  }

  uses community-count-top;
  uses as-path-length-top;
  uses bgp-match-set-conditions;
}

}

grouping community-set-config {
  description
    "Configuration data for BGP community sets";

  leaf community-set-name {
    type string;
    mandatory true;
    description
    "Community set name";
  }
}
"name / label of the community set -- this is used to reference the set in match conditions";

leaf-list community-member {
  type union {
    type oc-bgp-types:bgp-std-community-type;
    type oc-bgp-types:bgp-community-regexp-type;
    type oc-bgp-types:bgp-well-known-community-type;
  }
  description "members of the community set";
}

grouping community-set-state {
  description "Operational state data for BGP community sets";
}

grouping community-set-top {
  description "Top-level grouping for BGP community sets";

  container community-sets {
    description "Enclosing container for list of defined BGP community sets";

    list community-set {
      key "community-set-name";
      description "List of defined BGP community sets";

      leaf community-set-name {
        type leafref {
          path "../config/community-set-name";
        }
        description "Reference to list key";
      }

      container config {
        description "Configuration data for BGP community sets";

        uses community-set-config;
      }
    }
  }
}
container state {
  config false;

  description "Operational state data for BGP community sets";

  uses community-set-config;
  uses community-set-state;
}
}

grouping ext-community-set-config {
  description "Configuration data for extended BGP community sets";

  leaf ext-community-set-name {
    type string;
    description "name / label of the extended community set -- this is used to reference the set in match conditions";
  }

  leaf-list ext-community-member {
    type union {
      type oc-bgp-types:bgp-ext-community-type;
      type oc-bgp-types:bgp-community-regexp-type;
    }
    description "members of the extended community set";
  }
}

grouping ext-community-set-state {
  description "Operational state data for extended BGP community sets";
}

grouping ext-community-set-top {
  description "Top-level grouping for extended BGP community sets";

  container ext-community-sets {
    description "Enclosing container for list of extended BGP community sets";
  }
}
list ext-community-set {
  key "ext-community-set-name";
  description
      "List of defined extended BGP community sets";

leaf ext-community-set-name {
  type leafref {
    path ".../config/ext-community-set-name";
  }
  description
      "Reference to list key";
}

container config {
  description
      "Configuration data for extended BGP community sets";

  uses ext-community-set-config;
}

container state {
  config false;

  description
      "Operational state data for extended BGP community sets";

  uses ext-community-set-config;
  uses ext-community-set-state;
}

} }

grouping as-path-set-config {
  description
      "Configuration data for AS path sets";

leaf as-path-set-name {
  type string;
  description
      "name of the AS path set -- this is used to reference the set in match conditions";
}

leaf-list as-path-set-member {
  // TODO: need to refine typedef for AS path expressions
  type string;
}
description
    "AS path expression -- list of ASes in the set";
}
}

grouping as-path-set-state {
    description
        "Operational state data for AS path sets";
}

grouping as-path-set-top {
    description
        "Top-level grouping for AS path sets";

    container as-path-sets {
        description
            "Enclosing container for list of define AS path sets";

        list as-path-set {
            key "as-path-set-name";
            description
                "List of defined AS path sets";

            leaf as-path-set-name {
                type leafref {
                    path "./config/as-path-set-name";
                }
                description
                    "Reference to list key";
            }

            container config {
                description
                    "Configuration data for AS path sets";

                uses as-path-set-config;
            }

            container state {
                config false;
                description
                    "Operational state data for AS path sets";

                uses as-path-set-config;
                uses as-path-set-state;
            }
        }
    }
}

// augment statements

augment "/rpol:routing-policy/rpol: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";

    uses community-set-top;
    uses ext-community-set-top;
    uses as-path-set-top;
  }
}

grouping as-path-prepend-config {
  description "Configuration data for the AS path prepend action";

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

grouping as-path-prepend-state {
  description "Operational state data for the AS path prepend action";
}

grouping as-path-prepend-top {
  description "Top-level grouping for the AS path prepend action";

  container set-as-path-prepend {
    description "action to prepend local AS number to the AS-path a specified number of times";
  }
}
container config {
    description
        "Configuration data for the AS path prepend action";

    uses as-path-prepend-config;
}

container state {

    config false;
    description
        "Operational state data for the AS path prepend action";

    uses as-path-prepend-config;
    uses as-path-prepend-state;
}
}

grouping set-community-action-common {
    description
        "Common leaves for set-community and set-ext-community actions";

    leaf method {
        type enumeration {
            enum INLINE {
                description
                    "The extended communities are specified inline as a list";
            }
            enum REFERENCE {
                description
                    "The extended communities are specified by referencing a defined ext-community set";
            }
        }
        description
            "Indicates the method used to specify the extended communities for the set-ext-community action";
    }

    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.";
}
}


grouping set-community-inline-config {
  description
    "Configuration data for inline specification of set-community
    action";

  leaf-list communities {
    type union {
      type oc-bgp-types:bgp-std-community-type;
      type oc-bgp-types:bgp-well-known-community-type;
    }
    description
      "Set the community values for the update inline with
      a list.";
  }
}


grouping set-community-inline-state {
  description
    "Operational state data or inline specification of
    set-community action";
}

grouping set-community-inline-top {
  description
    "Top-level grouping or inline specification of set-community
    action";

  container inline {
    when "../config/method=INLINE" {
      description
        "Active only when the set-community method is INLINE";
    }
    description
      "Set the community values for the action inline with
      a list.";

    container config {
      description
        "Configuration data or inline specification of set-community
        action";

      uses set-community-inline-config;
    }
  }
}
container state {
  config false;
  description
    "Operational state data or inline specification of set-community action";
  uses set-community-inline-config;
  uses set-community-inline-state;
}
}

grouping set-community-reference-config {
  description
    "Configuration data for referencing a community-set in the set-community action";

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

grouping set-community-reference-state {
  description
    "Operational state data for referencing a community-set in the set-community action";
}

grouping set-community-reference-top {
  description
    "Top-level grouping for referencing a community-set in the set-community action";

  container reference {
    when "./.config/method=REFERENCE" {
      description
        "Active only when the set-community method is REFERENCES";
    }
    description

"Provide a reference to a defined community set for the set-community action";

container config {
  description
  "Configuration data for referencing a community-set in the set-community action";

  uses set-community-reference-config;
}

container state {
  config false;
  description
  "Operational state data for referencing a community-set in the set-community action";

  uses set-community-reference-config;
  uses set-community-reference-state;
}

grouping set-community-action-config {
  description
  "Configuration data for the set-community action";

  uses set-community-action-common;
}

grouping set-community-action-state {
  description
  "Operational state data for the set-community action";
}

grouping set-community-action-top {
  description
  "Top-level grouping for the set-community action";

  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).";
  }
}
container config {
    description
        "Configuration data for the set-community action";

    uses set-community-action-config;
}

container state {
    config false;

    description
        "Operational state data for the set-community action";

    uses set-community-action-config;
    uses set-community-action-state;
}

    uses set-community-inline-top;
    uses set-community-reference-top;
}
}

grouping set-ext-community-inline-config {
    description
        "Configuration data for inline specification of
        set-ext-community action";

    leaf-list communities {
        type union {
            type oc-bgp-types:bgp-ext-community-type;
            type oc-bgp-types:bgp-well-known-community-type;
        }

        description
            "Set the extended community values for the update inline
            with a list.";
    }
}

grouping set-ext-community-inline-state {
    description
        "Operational state data or inline specification of
        set-ext-community action";
}

grouping set-ext-community-inline-top {
    description
        "Top-level grouping or inline specification of set-ext-community
container inline {
  when "./.config/method=INLINE" {
    description
    "Active only when the set-community method is INLINE";
  }
  description
  "Set the extended community values for the action inline with a list.";
}

container config {
  description
  "Configuration data or inline specification of set-ext-community action";

  uses set-ext-community-inline-config;
}

container state {
  config false;

  description
  "Operational state data or inline specification of set-ext-community action";

  uses set-ext-community-inline-config;
  uses set-ext-community-inline-state;
}

grouping set-ext-community-reference-config {
  description
  "Configuration data for referencing a extended community-set in the set-ext-community action";

  leaf ext-community-set-ref {
    type leafref {
      path "/rpol:routing-policy/rpol:defined-sets/" +
        "oc-bgp-pol:bgp-defined-sets/" +
        "oc-bgp-pol:ext-community-sets/" +
        "oc-bgp-pol:ext-community-set/" +
        "oc-bgp-pol:ext-community-set-name";
    }
    description
    "References a defined extended community set by
grouping set-ext-community-reference-state {
  description
    "Operational state data for referencing an extended community-set in the set-ext-community action";
}

grouping set-ext-community-reference-top {
  description
    "Top-level grouping for referencing an extended community-set in the set-community action";

  container reference {
    when "../config/method=REFERENCE" {
      description
        "Active only when the set-community method is REFERENCE";
    }
    description
      "Provide a reference to an extended community set for the set-ext-community action";

    container config {
      description
        "Configuration data for referencing an extended community-set in the set-ext-community action";

        uses set-ext-community-reference-config;
    }

    container state {
      config false;

      description
        "Operational state data for referencing an extended community-set in the set-ext-community action";

      uses set-ext-community-reference-config;
      uses set-ext-community-reference-state;
    }
  }
}

grouping set-ext-community-action-config {
  description

"Configuration data for the set-ext-community action";

uses set-community-action-common;
}

grouping set-ext-community-action-state {
    description
    "Operational state data for the set-ext-community action";
}

grouping set-ext-community-action-top {
    description
    "Top-level grouping for the set-ext-community action";

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).";

    container config {
        description
        "Configuration data for the set-ext-community action";

        uses set-ext-community-action-config;
    }

    container state {

        config false;

        description
        "Operational state data for the set-ext-community action";

        uses set-ext-community-action-config;
        uses set-ext-community-action-state;
    }
}

    uses set-ext-community-inline-top;
    uses set-ext-community-reference-top;
}

}

grouping bgp-actions-config {
    description
    "Configuration data for BGP-specific actions";
leaf set-route-origin {
  type oc-bgp-types: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 update";
}

leaf set-next-hop {
  type bgp-next-hop-type;
  description "set the next-hop attribute in the route update";
}

leaf set-med {
  type bgp-set-med-type;
  description "set the med metric attribute in the route update";
}
}

grouping bgp-actions-state {
  description "Operational state data for BGP-specific actions";
}

grouping bgp-actions-top {
  description "Top-level grouping for BGP-specific actions";

  container bgp-actions {
    description "Top-level container for BGP-specific actions";

    container config {
      description "Configuration data for BGP-specific actions";

      uses bgp-actions-config;
    }

    container state {
      config false;
    }
  }
}
description
   "Operational state data for BGP-specific actions";

uses bgp-actions-config;
uses bgp-actions-state;
}
uses as-path-prepend-top;
uses set-community-action-top;
uses set-ext-community-action-top;
}
}
augment "/rpol:routing-policy/rpol:policy-definitions/" +
   "rpol:policy-definition/rpol:statements/rpol:statement/" +
   "rpol:conditions" {
   description
      "BGP policy conditions added to routing policy module";

   uses bgp-conditions-top;
}
augment "/rpol:routing-policy/rpol:policy-definitions/" +
   "rpol:policy-definition/rpol:statements/rpol:statement/" +
   "rpol:actions" {
   description
      "BGP policy actions added to routing policy module";

   uses bgp-actions-top;
}

// rpc statements

// notification statements

<CODE ENDS>

10. References

10.1. Normative references

         the Network Configuration Protocol (NETCONF)", RFC 6020,
         DOI 10.17487/RFC6020, October 2010,


10.2. Informative references

[I-D.ietf-rtgwg-policy-model]

[I-D.openconfig-netmod-opstate]

[I-D.ietf-netmod-routing-cfg]

[I-D.rtgyangdt-rtgwg-device-model]

[I-D.openconfig-rtgwg-network-instance]

[I-D.talwar-rtgwg-grpc-use-cases]
Specification, g., Kolhe, J., Shaikh, A., and J. George, "Use cases for gRPC in network management", draft-talwar-rtgwg-grpc-use-cases-00 (work in progress), July 2016.

Appendix A. 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, Jeff Haas, 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.
Appendix B. Change summary

B.1. Changes between revisions -01 and -02

- Refactored BGP model such that it is comprised of multiple sub-modules rather than independent modules.

- Remove the need for self-augmentation of the BGP model to allow the ability to import the model in wider structures more easily.

- Added new operational state values for BGP session established transitions and last-established timestamp. Also deprecated uptime operational state leaf.

- Added ability to select eligible paths for add-paths based on a policy.

B.2. Changes between revisions -00 and -01

- Updated module namespaces to reflect IETF standard namespace.

- Updated module filenames with ietf- prefix per RFC 6087 guidelines.

Authors’ Addresses

Anees Shaikh (editor)
Google
1600 Amphitheatre Pkwy
Mountain View, CA  94043
US

Email: aashaikh@google.com

Rob Shakir (editor)
Jive Communications, Inc.
1275 West 1600 North, Suite 100
Orem, UT  84057

Email: rjs@rob.sh