Interdomain Routing
Internet-Draft
Intended status: Informational
Expires: September 10, 2015

A. Shaikh
Google
K. D'Souza
AT&T
D. Bansal
Microsoft
R. Shakir
BT

March 9, 2015

BGP Model for Service Provider Networks
draft-shaikh-idr-bgp-model-01

Abstract

This document defines a YANG data model for configuring and managing BGP, including protocol, policy, and operational aspects based on 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 September 10, 2015.

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

1. Introduction

This document describes a YANG [RFC6020] data model for BGP [RFC4271] protocol and 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, however, to facilitate support from as large a set of routing hardware and software vendors as possible.

1.1. Goals and approach

This model does not (in the current iteration) aim to be feature complete (i.e., cover all possible features of a BGP implementation). Rather its development is driven by examination of BGP configurations in use across a number of operator network deployments.

The focus area of the first version of the model is on base BGP protocol configuration and policy configuration with "hooks" to add support for additional address families, as well as operational data to enable a common model for reading BGP-related state from devices.

The focus of the the BGP model described in this document is on providing the base 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 policies that relate to a neighbor - controlling the import and export of NLRI.

Configuration items that are deemed to be widely available in existing major BGP implementations are included in the model. Those configuration items that are only available from a single implementation are omitted from the model with the expectation they will be available in companion modules that augment the current model. This allows clarity in identifying data that is part of the vendor-neutral 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. Since implementations 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 but at a high level is organized into four 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].
- policy configuration -- configuration defining the policies that act on routes sent (received) to (from) peers or other routing protocols.
- operational state -- variables used for monitoring, management, etc. of BGP operations.

These modules also make use of the standard Internet types, such as IP addresses, 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 representing 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!
   +--rw global
   |  +-- (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 per-neighbor configuration being the most specific or lowest level, followed by peer-group, and finally global. Global configuration options reflect a subset of the per-peer-group or per-neighbor 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, the BGP bestpath 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 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 afi-safi
|     |  +--rw afi-safi* [afi-safi-name]
|     |     +--rw afi-safi-name              identityref
|     |     +--rw route-selection-options
|     |     +--rw use-multiple-paths!
|     |     +--rw apply-policy
|     |     +--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.shaikh-rtgwg-policy-model]. This model represents a condition-action policy framework. 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 then 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 parent entity.
- On a per-afi-safi basis within a neighbor or peer-group context, where the policy is specific to the AFI-SAFI.

```
+--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 data overview

The BGP operational model contains a set of parameters which relate 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 utilised 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) when within the peer-group context. To enable state to be added to particular contexts, the tree is augmented through the base BGP module to add these variables, without requiring separate groupings.

3. 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.

4. IANA Considerations

This YANG data model and the component modules currently use a temporary ad-hoc namespace. If and when it is placed on redirected for the standards track, 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].

5. YANG modules

The modules comprising the BGP configuration and operational model are described by the YANG modules in the sections below. The base module imports the other modules to create the overall model.

5.1. BGP base items

```
<CODE BEGINS> file bgp.yang
module bgp {

  yang-version "1";

  // namespace
  namespace "http://openconfig.net/yang/bgp";

  prefix "bgp";

  // import some basic inet types
  import ietf-inet-types { prefix inet; }
  import bgp-multiprotocol { prefix bgp-mp; }
  import routing-policy { prefix rpol; }
  import bgp-types { prefix bgp-types; }
  import bgp-operational { prefix bgp-op; }

  // 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 ]
          +-> peer group
            +-> [ peer group config ]
            +-> AFI / SAFI [ per-AFI overrides ]
          +-> neighbor
            +-> [ neighbor config ]
            +-> [ optional pointer to peer-group ]
            +-> AFI / SAFI [ per-AFI overrides ]";

    revision "2014-03-05" {
        description
          "";
        reference "TBD";
    }

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

      leaf as {
        type inet:as-number;
        mandatory "true";
        description
          "Local autonomous system number of the router.  Uses
          the 32-bit as-number type from the model in RFC 6991.";
      }
      leaf router-id {
        type inet:ipv4-address;
        description
          "Router id of the router, expressed as an
          32-bit value, IPv4 address.";
      }
    }
```

```
    grouping bgp-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-confederation_config {
      description
        "Configuration options specifying parameters when the local
        router is within an autonomous system which 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-neighbor_config {
      description
        "Neighbor level configuration items.";
```

```
        leaf peer-as {
          type inet:as-number;
          mandatory "true";
          description
            "AS number of the peer.";
        }

        leaf peer-type {
            type 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 bgp-types:remove-private-as-option;
          description
            "Remove private AS numbers from updates sent to peers.";
        }

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

        leaf send-community {
          type 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-neighbor-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
          "Mininum time interval in seconds between transmission
          of BGP updates to neighbors";
        reference
          "RFC 4271 - A Border Gateway Protocol 4, Sec 10";
      }

      leaf send-update-delay {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Time interval between routes changing in the routing
          table and corresponding updates sent to neighbors --
          serves to batch updates";
      }
    }

    grouping bgp-neighbor-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;
        description
          "Turns path mtu discovery for BGP TCP sessions on (true)
          or off (false)";
      }

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

      leaf local-address {
        type inet:ip-address;
        description
```

```
          "Set the local IP (either IPv4 or IPv6) address to use
          for the session when sending BGP update messages.";
      }
    }

    grouping bgp-neighbor-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 reated as though the NLRI is
          withdrawn - avoiding session reset";
        reference "draft-ietf-idr-error-handling-16";
      }
    }

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

      leaf multihop-ttl {
        type uint8;
        default 1;
        description
          "Time-to-live for multihop BGP sessions.  The default
          value of 1 is for directly connected peers (i.e.,
          multihop disabled";
      }
```

```
    }

    grouping bgp-neighbor-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 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-neighbor-as-path-options_config {
      description
        "Configuration parameters allowing manipulation of the AS_PATH
        attribute";

      leaf allow-own-as {
        // rjs: this could be uint32, but ALU SROS treats as a
        //      boolean. JUNOS, IOS & IOS XR treat as an integer
        //      specifying the number of occurrences.
        type boolean;
        default "false";
        description
          "Specify whether routes for which the local router's AS
          appears in the path are rejected as looped.";
      }

      leaf replace-peer-as {
        type boolean;
        default "false";
        description
          "Replace occurrences of the peer's AS in the AS_PATH
          with the local autonomous system number";
      }
    }
```

```
    grouping bgp-neighbor-add-paths_config {
      description
        "Configuration parameters specfying whether the local system
        will send or receive multiple paths using ADD_PATHS";

      leaf receive {
        type empty;
        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";
      }
    }

    grouping bgp-neighbor-peer-group_config {
      description
        "Configuration parameters indicating whether the specified peer
        is to be considered as part of a peer-group - and therefore
        inherit its configuration";

      leaf peer-group {
        type leafref {
          // we are at /bgp/neighbors/neighbor/
          path "/bgp/peer-groups/peer-group/peer-group-name";
          require-instance true;
        }
        description
          "The peer-group with which this neighbor is associated";
      }
    }

    grouping bgp-graceful-restart_config {
      description
        "Configures BGP graceful restart, which is a negotiated
        option that indicates that a BGP speaker is able to retain
        forwarding state when a BGP session restarts";

      reference "RFC 4724: Graceful Restart Mechanism for BGP";
      container graceful-restart {
        presence
          "Presence of this item indicates that BGP graceful restart
          is enabled.";
```

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

    grouping bgp-neighbor-graceful-restart_config {
      description
        "Configuration parameters relating to BGP graceful restart.";

      leaf restart-time {
        type uint16 {
          range 0..4096;
        }
        description
          "Estimated time in seconds for the BGP session to be
          re-established after a restart.  This is a 12-bit value
          advertised by the router to peers.  Per RFC 4724, the
          suggested default value is <= the hold-time value";
      }

      leaf stale-routes-time {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Sets an upper bound on the time in seconds that stale
          routes will be retained by the router after a session is
          restarted";
      }
    }

    // ***********************************************************
    // *              configuration context containers           *
    // ***********************************************************

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

      container confederation {
        presence
          "Presence of this container indicates that the local AS is
          part of a 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-confederation_config;
```

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

      uses bgp-mp:bgp-use-multiple-paths_config;
      uses bgp-graceful-restart_config;

      container afi-safi {
        description
          "Address family specific configuration";
        uses bgp-mp:bgp-global-afi-safi-list;
      }
    }

    grouping bgp-neighbors {
      description
        "BGP neighbors configured on the local system";
      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 inet:ip-address;
          description
            "Address of the BGP peer, either in IPv4 or IPv6";
        }
        uses bgp-neighbor-group;
      }
    }

    grouping bgp-peer-group {
      description
        "BGP peer-groups configured on the local system";
      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 string;
```

```
          description
            "";
        }
        uses bgp-neighbor-group;
      }
    }

    grouping bgp-neighbor-group {
      description
        "Parameters related to a BGP neighbor or group";

      container config {
        description
          "Configuration parameters relating to the BGP neighbor or
          group";
        uses bgp-neighbor_config;
      }
      container state {
        config false;
        description
          "State information relating to the BGP neighbor or group";
        uses bgp-neighbor_config;
      }

      container timers {
        description
          "Timers related to a BGP neighbor or group";
        container config {
          description
            "Configuration parameters relating to timers used for the
            BGP neighbor or group";
          uses bgp-neighbor-timers_config;
        }
        container state {
          config false;
          description
            "State information relating to the timers used for the BGP
            neighbor or group";
          uses bgp-neighbor-timers_config;
        }
      }

      container transport {
        description
          "Transport session parameters for the BGP neighbor or group";
        container config {
          description
            "Configuration parameters relating to the transport
```

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

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

      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 neighbor or group";
          uses bgp-neighbor-logging-options_config;
        }
        container state {
          config false;
          description
            "State information relating to logging for the BGP neighbor
            or group";
          uses bgp-neighbor-logging-options_config;
```

```
        }
      }

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

      container route-reflector {
        description
          "Route reflector parameters for the BGP neighbor or group";
        container config {
          description
            "Configuraton parameters relating to route reflection
            for the BGP neighbor or group";
          uses bgp-neighbor-route-reflector_config;
        }
        container state {
          config false;
          description
            "State information relating to route reflection for the
            BGP neighbor or group";
          uses bgp-neighbor-route-reflector_config;
        }
      }

      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-neighbor-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-neighbor-as-path-options_config;
        }
      }

      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-neighbor-add-paths_config;
        }
        container state {
          config false;
          description
            "State information associated with ADD_PATHS";
          uses bgp-neighbor-add-paths_config;
        }
      }

      container afi-safi {
        description
          "Per-address-family configuration parameters associated with
          the neighbor or group";
        uses bgp-mp:bgp-global-afi-safi-list;
      }

      uses bgp-graceful-restart_config;

      uses rpol:apply-policy-group;
    }

    // add peer-group pointer only for the neighbor list
    augment /bgp/neighbors/neighbor/config {
      description
        "Augmentation to allow association of a neighbor with a
        peer-group";
      uses bgp-neighbor-peer-group_config;
    }

    augment /bgp/neighbors/neighbor/state {
```

```
      description
        "Augmentation to reflect the association of a neighbor with a
        peer-group";
      uses bgp-neighbor-peer-group_config;
    }

    augment /bgp/peer-groups/peer-group {
      description
        "Augmentation to add multipath configuration to a peer-group";
      uses bgp-mp:bgp-use-multiple-paths_config;
    }

    augment /bgp/neighbors/neighbor {
      description
        "Augmentation to add the multipath configuration to a
        neighbor";
      uses bgp-mp:bgp-use-multiple-paths-neighbor_config;
    }

    // ***********************************************************
    // *              Augmentations to add state                 *
    // *  (rjs: cleaner to have these in the base module to avoid *
    //     needing to specify which module - e.g. augment of     *
    //     /bgp:bgp/bgp:neighbors/bgp:neighbor...)               *
    // ***********************************************************
    augment /bgp/neighbors/neighbor/state {
      description
        "Augmentation to add operational state related to a particular
        BGP neighbor";
      uses bgp-op:bgp-neighbor_state;
    }

    augment /bgp/neighbors/bgp:neighbor/state {
      description
        "Augmentation to add operational state related to a particular
        BGP neighbor";

      container messages {
        description
          "Counters for BGP messages sent and received from the
          neighbor";
        container sent {
          description
            "Counters relating to BGP messages sent to the neighbor";
          uses bgp-op:bgp-neighbor-message-counters-sent_state;
          }

        container received {
```

```
          description
            "Counters for BGP messages received from the neighbor";
          uses bgp-op:bgp-neighbor-message-counters-received_state;
        }
      }

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

    augment /bgp:bgp/bgp:neighbors/neighbor/timers/state {
      description
        "Augmentation to add the operational state of timers associated
        with the BGP neighbor";
      uses bgp-op:bgp-neighbor-timers_state;
    }

    augment /bgp/neighbors/neighbor/transport/state {
      description
        "Augmentation to add the operational state of the transport
        session associated with the BGP neighbor";
      uses bgp-op:bgp-neighbor-transport_state;
    }

    augment /bgp/neighbors/neighbor/error-handling/state {
      description
        "Augmentation to add the operational state of the error
        handling associated with the BGP neighbor";
      uses bgp-op:bgp-neighbor-error-handling_state;
    }

    augment /bgp/neighbors/neighbor/graceful-restart/state {
      description
        "Augmentation to add the operational state of graceful-restart
        associated with a BGP neighbor";
      uses bgp-op:bgp-afi-safi-graceful-restart_state;
    }

    augment /bgp/peer-groups/peer-group/state {
      description
        "Augmentation to add the operational state and counters
        relating to a BGP peer-group";
      uses bgp-op:bgp-peer-group_state;
    }
```

```
    augment /bgp/global/afi-safi/afi-safi/state {
      description
        "Augmentation to add operational state and counters
        on a per-AFI-SAFI basis to the global BGP router";
      uses bgp-op:bgp-global-afi-safi_state;
    }

    augment /bgp/neighbors/neighbor/afi-safi/afi-safi/state {
      description
        "Augmentation to add per-AFI-SAFI operational state
        and counters to the BGP neighbor";
      uses bgp-op:bgp-neighbor-afi-safi_state;
    }

    // ***********************************************************
    // *              module structure containers                *
    // ***********************************************************

    container bgp {
      presence "Container for BGP protocol hierarchy";
      description
        "Top-level configuration and state for the BGP router";

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

      container neighbors {
        description
          "Configuration for BGP neighbors";
        uses bgp-neighbors;
      }

      container peer-groups {
        description
          "Configuration for BGP peer-groups";
        uses bgp-peer-group;
      }
    }
  }
  <CODE ENDS>
```

5.2. BGP base types

```
<CODE BEGINS> file bgp-types.yang
module bgp-types {
    yang-version "1";

    namespace "http://openconfig.net/yang/bgp-types";

    prefix "bgp-types";

    import ietf-inet-types { prefix inet; }
    import ietf-yang-types { prefix yang; }

    // 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";

  revision "2015-03-03" {
    description "Initial revision";
    reference "TBD";
  }

  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";
  }


  typedef remove-private-as-option {
    type enumeration {
```

```
        enum ALL {
          description "remove all private ASes in the path";
        }
        enum REPLACE {
          description "replace private ASes with local AS";
        }
      }
      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";
    }

    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";
      reference "draft-ietf-idr-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-labelled-unicast {
      base afi-safi-type;
      description
        "Labelled IPv4 unicast (AFI,SAFI = 1,4)";
      reference "RFC3107";
    }

    identity ipv6-labelled-unicast {
      base afi-safi-type;
      description
        "Labelled 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)";
    }

    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";
    }


  }
  <CODE ENDS>
```

5.3. BGP policy items

```
  <CODE BEGINS> file bgp-policy.yang
  module bgp-policy {
```

```
yang-version "1";

// namespace
namespace "http://openconfig.net/yang/bgp-policy";

prefix "bgp-pol";

// import some basic types
import ietf-inet-types { prefix inet; }
import routing-policy {prefix rpol; }
import policy-types {prefix pt; }



// meta
organization
  "OpenConfig working group";

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

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

revision "2014-11-30" {
  description
    "Updated model to augment base routing-policy module";
  reference "TBD";
}

// extension statements

// feature statements

// identity statements

// typedef statements

typedef bgp-as-path-prepend-repeat {
  type uint8;
  description
    "Option for the BGP as-prepend policy action.  Prepends the
    local AS number repeated n times";
}
```

```
    typedef bgp-well-known-community-type {
      type enumeration {
        enum INTERNET {
          description "entire Internet community (0x00000000)";
        }
        enum NO_EXPORT {
          // value 0xFFFFFF01;
          description "no export";
        }
        enum NO_ADVERTISE {
          description "no advertise (0xFFFFFF02)";
        }
        enum NO_EXPORT_SUBCONFED {
          description "no export subconfed, equivalent to
          local AS (0xFFFFFF03)";
        }
      }
      description
        "Type definition for well-known IETF community attribute
        values";
      reference "RFC 1997 - BGP Communities Attribute";
    }


    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..0xFFFEFFFF
        }
        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 string {
        pattern '([0-9\.]+(:[0-9]+)?:[0-9]+)';
      }
      description
        "Type definition for extended community attributes";
      reference "RFC 4360 - BGP Extended Communities Attribute";
    }

    typedef bgp-community-regexp-type {
      // TODO: needs more work to decide what format these regexps can
      // take.
      type string;
      description
        "Type definition for communities specified as regular
        expression patterns";
    }

    typedef bgp-origin-attr-type {
      type enumeration {
        enum IGP {
          value 0;
          description "Origin of the NLRI is internal";
        }
        enum EGP {
          value 1;
          description "Origin of the NLRI is EGP";
        }
        enum INCOMPLETE {
          value 2;
          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 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";
        }
        enum NULL {
          description "set the community attribute to empty / NULL";
        }
      }
      description
        "Type definition for options when setting the community
        attribute in a policy action";
    }

    typedef bgp-next-hop-type {
      type union {
        type inet:ip-address;
        type enumeration {
          enum SELF {
            description "special designation for local router's own
            address, i.e., next-hop-self";
          }
        }
      }
      description "type definition for specifying next-hop in policy
      actions";
    }

    typedef bgp-set-med-type {
      type union {
        type uint32;
        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";
    }

    // grouping statements

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

```
        defined set";

      container match-community-set {
        presence
          "The presence of this container indicates that the routes
          should match the referenced community-set";

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

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

      container match-ext-community-set {
        presence
          "The presence of this container indicates that the routes
          should match the referenced extended community set";

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

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

      container match-as-path-set {
        presence
          "The presence of this container indicates that the route
```

```
          should match the referenced as-path set";

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

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

    grouping bgp-attribute-conditions {
      description
        "Condition statement definitions for comparing a BGP route
        attribute to a specified value";

      leaf med-eq {
        type uint32;
        description
          "Condition to check if the received MED value is equal to
          the specified value";
      }

      leaf origin-eq {
        type 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;
        description
          "List of next hop addresses to check for in the route
          update";
      }

      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";
      }

      container community-count {

        presence "node is present in the config data to indicate a
        community-count condition";

        description
          "Value and comparison operations for conditions based on the
          number of communities in the route update";

        uses pt:attribute-compare-operators;

      }

      container as-path-length {

        presence "node is present in the config data to indicate a
        as-path-length condition";

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

        uses pt:attribute-compare-operators;
      }

      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";
      }
    }


    // 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";

         list community-set {
           key community-set-name;
           description
               "Definitions for community sets";

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

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

         }

         list ext-community-set {
           key ext-community-set-name;
           description
               "Definitions for extended 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-members {
             type union {
               type bgp-ext-community-type;
```

```
            // TODO: is regexp support needed for extended
            // communities?
            type bgp-community-regexp-type;
          }
          description
              "members of the extended community set";
        }
      }

      list as-path-set {
        key as-path-set-name;
        description
            "Definitions for AS path sets";

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

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

      }
    }
  }

  augment "/rpol:routing-policy/rpol:policy-definition/" +
    "rpol:statement/rpol:conditions" {
    description "BGP policy conditions added to routing policy
    module";

    container bgp-conditions {
      description "Policy conditions for matching
      BGP-specific defined sets or comparing BGP-specific
      attributes";

      uses bgp-match-conditions;
      uses bgp-attribute-conditions;
    }
  }

  augment "/rpol:routing-policy/rpol:policy-definition/" +
```

```
       "rpol:statement/rpol:actions" {
       description "BGP policy actions added to routing policy
       module";

       container bgp-actions {
         description
           "Definitions for policy action statements that
           change BGP-specific attributes of the route";

         container set-as-path-prepend {

           presence "node is pesent in the config data to use the AS
         prepend action";
           description
               "action to prepend local AS number to the AS-path a
           specified number of times";

           leaf repeat-n {
             type uint8;
             description "number of times to prepend the local AS
             number";
           }
         }

         container set-community {
           presence "node is present in the config data when
           set-community action is used";
           description
             "action to set the community attributes of the route, along
             with options to modify how the community is modified";

           leaf-list communities {
             type union {
               type bgp-std-community-type;
               type bgp-well-known-community-type;
             }
             description
                 "community values for the update";
           }

           leaf options {
             type bgp-set-community-option-type;
             description
               "options for modifying the community attribute with the
               specified values";
           }
         }
```

```
        container set-ext-community {

          presence "node is present in the config data when
          set-community action is used";
          description
            "action to set the extended community attributes of the
            route, along with options to modify how the community is
            modified";

          leaf-list communities {
            type union {
              type bgp-ext-community-type;
              type bgp-well-known-community-type;
            }
            description
                "community values for the update";
          }

          leaf options {
            type bgp-set-community-option-type;
            description
              "options for modifying the community attribute with the
              specified values";
          }
        }

        leaf set-route-origin {
          type 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";
        }
```

```
      }
    }

    // rpc statements

    // notification statements
  }
  <CODE ENDS>
```

5.4. BGP multiprotocol items

```
  <CODE BEGINS> file bgp-multiprotocol.yang
  module bgp-multiprotocol {

    yang-version "1";

    // namespace
    namespace "http://openconfig.net/yang/bgp-multiprotocol";

    prefix "bgp-mp";

    // import some basic inet types
    import routing-policy { prefix rpol; }
    import bgp-types { prefix bgp-types; }
    import bgp-operational { prefix bgp-op; }

    // meta
    organization
      "OpenConfig working group";

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

    description
      "This module is part of a YANG model for BGP protocol
      configuration, focusing on configuration of multiprotocol
      BGP, in particular various relevant address families (AFI) and
      sub-address families (SAFI).

      Identities (rather than enumerated types) are used to identify
      each AFI / SAFI type to make it easier for users to extend to
      pre-standard or custom AFI/SAFI types.  This module is only
      intended to capture the most";


    revision "2014-11-30" {
```

```
      description
        "Refactored multiprotocol module";
      reference "TBD";
    }

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

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

        presence
          "Presence of this container indicates that the IPv4 Unicast
          AFI,SAFI is enabled for a neighbour or group";

        description "IPv4 unicast configuration options";

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

        // placeholder for IPv4 unicast  specific configuration
      }
    }

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

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

        presence
          "Presence of this container indicates that the IPv6 Unicast
          AFI,SAFI is enabled for a neighbour or group";

        description "IPv6 unicast configuration options";

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

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

    grouping ipv4-labelled-unicast-group {
      description
        "Group for IPv4 Labelled Unicast configuration options";

      container ipv4-labelled-unicast {
        when "../afi-safi-name = 'bgp-mp:ipv4-labelled-unicast'" {
          description
            "Include this container for IPv4 Labelled Unicast specific
            configuration";
        }

        presence
          "Presence of this container indicates that the IPv4 Labelled
          Unicast AFI,SAFI is enabled for a neighbour or group";

        description "IPv4 Labelled Unicast configuration options";

        uses all-afi-safi-common;

        // placeholder for IPv4 Labelled Unicast specific config
        // options
      }
    }

    grouping ipv6-labelled-unicast-group {
      description
        "Group for IPv6 Labelled Unicast configuration options";

      container ipv6-labelled-unicast {
        when "../afi-safi-name = 'bgp-mp:ipv6-labelled-unicast'" {
          description
            "Include this container for IPv6 Labelled Unicast specific
            configuration";
        }

        presence
          "Presence of this container indicates that the IPv6 Labelled
          Unicast AFI,SAFI is enabled for a neighbour or group";

        description "IPv6 Labelled Unicast configuration options";

        uses all-afi-safi-common;
```

```
        // placeholder for IPv6 Labelled Unicast specific config
        // options.
      }
    }

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

      container l3vpn-ipv4-unicast {
        when "../afi-safi-name = 'bgp-mp:l3vpn-ipv4-unicast'" {
          description
            "Include this container for IPv4 Unicast L3VPN specific
            configuration";
        }

        presence
          "Presence of this container indicates that IPv4 Unicast L3VPN
          AFI,SAFI is enabled for a neighbour or group";

        description "Unicast IPv4 L3VPN configuration options";

        // include common L3VPN configuration options
        uses l3vpn-ipv4-ipv6-unicast-common;

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

    grouping l3vpn-ipv6-unicast-group {
      description
        "Group for IPv6 Unicast L3VPN configuration options";

      container l3vpn-ipv6-unicast {
        when "../afi-safi-name = 'bgp-mp:l3vpn-ipv6-unicast'" {
          description
            "Include this container for unicast IPv6 L3VPN specific
            configuration";
        }

        presence "Presence of this container indicates that the IPv6
                  Unicast L3VPN AFI,SAFI is enabled";

        description "Unicast IPv6 L3VPN configuration options";

        // include common L3VPN configuration options
        uses l3vpn-ipv4-ipv6-unicast-common;
```

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

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

      container l3vpn-ipv4-multicast {
        when "../afi-safi-name = 'bgp-mp:l3vpn-ipv4-multicast'" {
          description
            "Include this container for multicast IPv6 L3VPN specific
            configuration";
        }

        presence "Presence of this container indicates the
                  IPv4 L3VPN Unicast AFI-SAFI is enabled";

        description "Multicast IPv4 L3VPN configuration options";

        // include common L3VPN multicast options
        uses l3vpn-ipv4-ipv6-multicast-common;

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

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

      container l3vpn-ipv6-multicast {
        when "../afi-safi-name = 'bgp-mp:l3vpn-ipv6-multicast'" {
          description
            "Include this container for multicast IPv6 L3VPN specific
            configuration";
        }

        presence
          "Presence of this container indicates that the IPv6 Multicast
          L3VPN AFI,SAFI is enabled";

        description "Multicast IPv6 L3VPN configuration options";

        // include common L3VPN multicast options
        uses l3vpn-ipv4-ipv6-multicast-common;
```

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

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

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

        presence
          "Presence of this container indicates that the BGP-signalled
          VPLS AFI,SAFI is enabled";

        description "BGP-signalled VPLS configuration options";

        // include common L2VPN options
        uses l2vpn-common;

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

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

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

        presence
          "Presence of this container indicates that the BGP EVPN
          AFI,SAFI is enabled";

        description "BGP EVPN configuration options";

        // include common L2VPN options
```

```
        uses l2vpn-common;

        // placeholder for BGP EVPN specific configuration options
      }
    }



    grouping bgp-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 empty;
        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";
      }
    }

    grouping bgp-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-use-multiple-paths-ebgp_config {
      description
        "Configuration parameters relating to multipath for eBGP";

      uses bgp-use-multiple-paths-ebgp-as-options_config;

      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-use-multiple-paths-ibgp_config {
```

```
      description
        "Configuration parmaeters 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-use-multiple-paths_config {
      description
        "Configuration parameters relating to multipath for BGP - both
        iBGP and eBGP";

      container use-multiple-paths {
        presence
          "Presence of this container indicates that multipath is
          enabled for eBGP and iBGP. Absence of the container
          indicates that multipath is disabled";

        description
          "Parameters related to the use of multiple paths for the
          same NLRI";

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

        container ibgp {
          description
            "Multipath parameters for iBGP";
          container config {
            description
```

```
            "Configuration parameters relating to iBGP multipath";
          uses bgp-use-multiple-paths-ibgp_config;
        }
        container state {
          config false;
          description
            "State information relating to iBGP multipath";
          uses bgp-use-multiple-paths-ibgp_config;
        }
      }
    }
  }

  grouping bgp-use-multiple-paths-neighbor_config {
    description
      "Per-neighbor configuration for multipath for BGP";

    container use-multiple-paths {
      presence
        "Presence of this container indicates that multiple paths
        from this neighbor should be installed into the RIB. Absence
        of this container results in the multipath configuration
        being inherited from the peer-group if it exists.";

      description
        "Parameters related to the use of multiple-paths for the same
        NLRI when they are received only from this neighbor";

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

  grouping bgp-afi-safi_config {
    description
```

```
        "Configuration parameters used for all BGP AFI-SAFIs";

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

    grouping 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 bgp-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 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)";
      }
    }

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

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

      // include common afi-safi options.
      uses 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-SAFI options";
        uses ipv4-ipv6-unicast-common_config;
      }
```

```
      container state {
        config false;
        description
          "State information for common IPv4 and IPv6 unicast
          parameters";
        uses ipv4-ipv6-unicast-common_config;
      }
    }

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

      // placeholder -- specific configuration options that are generic
      // across IPv[46] unicast address families.
      uses all-afi-safi-common;
    }

    grouping 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 all-afi-safi-common;
    }

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

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

    // *********** STRUCTURE GROUPINGS **********************

    grouping bgp-global-afi-safi-list {
      description
        "List of address-families associated with the BGP instance,
        a peer-group or neighbor";

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

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

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

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

        uses bgp-use-multiple-paths_config;

        container config {
          description
            "Configuration parameters for the AFI-SAFI";
          uses bgp-afi-safi_config;
        }
        container state {
          config false;
          description
            "State information relating to the AFI-SAFI";
          uses bgp-afi-safi_config;
          uses bgp-op:bgp-afi-safi_state;
        }

        // import and export policy included for the afi/safi
        uses rpol:apply-policy-group;

        uses ipv4-unicast-group;
        uses ipv6-unicast-group;
```

```
        uses ipv4-labelled-unicast-group;
        uses ipv6-labelled-unicast-group;
        uses l3vpn-ipv4-unicast-group;
        uses l3vpn-ipv6-unicast-group;
        uses l3vpn-ipv4-multicast-group;
        uses l3vpn-ipv6-multicast-group;
        uses l2vpn-vpls-group;
        uses l2vpn-evpn-group;
      }
    }
  }
  <CODE ENDS>
```

5.5. BGP operational data items

```
  <CODE BEGINS> file bgp-operational.yang
  module bgp-operational {

    yang-version "1";

    // namespace
    // TODO: change to an ietf or other more generic namespace
    namespace "http://openconfig.net/yang/bgp-operational";

    prefix "bgp-op";

    // import some basic inet types
    import ietf-inet-types { prefix inet; }
    import ietf-yang-types { prefix yang; }
    import bgp-types { prefix bgp-types; }

    // meta

    organization
      "OpenConfig working group";

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

    description
      "This module is part of a YANG model for BGP protocol
      configuration, focusing on operational data (i.e., state
      variables) related to BGP operations";


    revision "2014-12-02" {
```

```
      description
        "Initial revision";
      reference "TBD";
    }

    // extension statements

    // feature statements

    // identity statements

    // typedef statements

    // grouping statements

    grouping bgp-counters-message-types_common {
      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-context-pfx-path-counters_common {
      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
          "";
      }
    }

    grouping bgp-global_state {
      description
        "Grouping containing operational parameters relating to the
        global BGP instance";
      uses bgp-context-pfx-path-counters_common;
    }

    grouping bgp-global-afi-safi_state {
      description
        "Grouping containing operational parameters relating to each
        AFI-SAFI within the BGP global instance";
      uses bgp-context-pfx-path-counters_common;
    }

    grouping bgp-peer-group_state {
      description
        "Grouping containing operational parameters relating to a BGP
        peer group";
      uses bgp-context-pfx-path-counters_common;
    }

    grouping bgp-neighbor_state {
      description
        "Grouping containing operational state variables relating 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 conncetion 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-list supported-capabilities {
        type identityref {
          base bgp-types:bgp-capability;
        }
        description
          "BGP capabilities negotiated as supported with the peer";
      }
    }

    grouping bgp-neighbor-afi-safi_state {
      description
        "Operational state on a per-AFI-SAFI basis for a BGP
        neighbor";

        uses bgp-neighbor-prefix-counters_state;
    }

    grouping bgp-neighbor-prefix-counters_state {
      description
        "Counters for BGP neighbor sessions";

      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-message-counters-sent_state {
      description
        "Counters relating to messages sent to a BGP neighbor";
      uses bgp-counters-message-types_common;
    }

    grouping bgp-neighbor-message-counters-received_state {
      description
        "Counters relating to the mesages received from a BGP
        neighbor";
      uses bgp-counters-message-types_common;
    }

    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 port being used by the peer for the TCP session
          supporting the BGP session";
      }

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

    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 uptime {
        type yang:timeticks;
```

```
        description
          "This timer determines the amount of time since the
          BGP last transitioned in or out of the Established
          state";
      }

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

    grouping bgp-afi-safi_state {
      description
        "Operational state information relevant to all address
        families that may be carried by the BGP session";

      // placeholder - options in this container are
      // valid in both the global and per-neighbor
      // paths

    }

    grouping bgp-afi-safi-graceful-restart_state {
      description
        "Operational state information relevant to graceful restart
        for BGP";

      leaf active {
        type boolean;
        description
          "Whether graceful-restart has been enabled for the AFI,
          SAFI for the peer";
      }

      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";
      }
    }
```

```
}

<CODE ENDS>
```

6. References

6.1. Normative references

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, January 2007.

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

6.2. Informative references

[I-D.openconfig-netmod-opstate]
Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-00 (work in progress), March 2015.

[I-D.shaikh-rtgwg-policy-model]
Shaikh, A., Shakir, R., D'Souza, K., and C. Chase, "Routing Policy Configuration Model for Service Provider Networks", draft-shaikh-rtgwg-policy-model-00 (work in progress), January 2015.

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, Jeff Haas, Ina Minei, Carl Moberg, Ashok Narayanan, Einar Nilsen-Nygaard, Keyur Patel, Adam Simpson, Puneet Sood, Jason Sterne, and Jim Uttaro.

Appendix B. Change summary

B.1. Changes between revisions -00 and -01

The -01 revision reflects a number of changes, many based on feedback from implementors of the model on various routing platforms.

- Refactored model to explicitly provide 'config' and 'state' containers at each leaf node to enable consistent and predictable access to operational state data corresponding to configuration data. This is based on the model design in [I-D.openconfig-netmod-opstate].

- Refactored multiprotocol module with explicit set of supported AFI-SAFI combinations (using YANG identities) in a flattened list. Focus was on common config with more AFI-SAFI specific configuration forthcoming in future revisions.

- Refactored BGP policy module to work with a new general routing policy model [I-D.shaikh-rtgwg-policy-model] by augmenting it with BGP-specific policy options (conditions, actions, and defined sets).

- Added enclosing containers to lists (e.g., neighbors, peer-groups, and AFI-SAFI)

- Removed neighbor configuration from the peer-group hierarchy. Neighbor configuration now has a peer-group leaf which references the peer group to which the neighbor belongs.

- Several new configuration items added to base bgp module, including adding some configuration items to the global hierarchy level.

Authors' Addresses

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

Email: aashaikh@google.com

Kevin D'Souza
AT&T
200 S. Laurel Ave
Middletown, NJ
US

Email: kd6913@att.com

Deepak Bansal
Microsoft
205 108th Ave. NE, Suite 400
Bellevue, WA
US

Email: dbansal@microsoft.com

Rob Shakir
BT
pp. C3L, BT Centre
81, Newgate Street
London EC1A 7AJ
UK

Email: rob.shakir@bt.com
URI: http://www.bt.com/