Network Working Group M. Pels
Internet-Draft RIPE NCC
Intended status: Standards Track ~~Informational~~
19 August 2024
Expires: 20 February 2025

A YANG ~~Module~~ Data Model for BGP Communities
draft-ietf-grow-yang-bgp-communities-02

Abstract

This document defines a YANG data model for the structured specification of BGP communities. The model provides operators with a way to publish their locally defined BGP communities in a ~~standardised~~standardized format.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on 20 February 2025.

Table of Contents

Commenté [MB1]: Is more appropriate for this spec

Commenté [MB2]: For consistency with other uses in the doc

1. Introduction

ISPs ~~operators~~ use BGP communities [RFC1997][RFC4360][RFC8092] to add information to their prefix announcements or to let customers influence routing behaviour inside the network of the ISP. Each ISP defines for itself which BGP communities to support and how the structure of these communities should be interpreted.

This document provides a YANG [RFC7950] module for describing the structure and meaning of BGP communities, Extended BGP communities, and Large BGP communities. ISPs ~~operators~~ can use this standardized format to publish their community definitions ~~in a standardised format~~. Section 3 elaborates on further advantages of using such a standardized format.

Commenté [MB3]: I'm afraid the module used in this document is not for management purposes.

I think the approach in RFC 8791 - YANG Data Structure Extensions is more appropriate here.

Commenté [MB4]: Add refs for each of those

2. Terminology

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

The meanings of the symbols in tree diagrams are defined in [RFC8340].

3. Rationale

ISPs ~~operators~~ may define various BGP communities that have local significance ~~inside of~~ within their network. These communities could be used to add miscellaneous information to a prefix announcement. For example, a community "64501:1:528" may signify that the AS with ASN AS64501 is originating a prefix from a point of presence in The Netherlands (ISO 3166-1 code 528). ~~Alternatively, c~~Communities could be used to allow customers of an ISP to control the routing behavior of their prefixes inside the ISP. For example, a community "64501:4:64498" attached to a prefix advertised to AS64501 by a customer may be interpreted by AS64501 to mean that this prefix must not be propagated to AS64498.

For both use cases it is necessary for the ISP to communicate the meaning of their locally defined communities to others. Currently this is typically done by publishing a list of communities on a web page, or as a remark inside an 'autnum' object in the Internet Routing Registry. This makes it cumbersome to determine ~~if~~ whether and where an ISP publishes community information. The lack of a well-defined

Commenté [MB5]: How this is solved by this doc?

The uri in the model is not sufficient as one has to find where to find this data at the first place.

structure makes it hard to develop tools for automatically parsing community information and eventually trigger configuration actions.

The purpose of the YANG model defined in this document is to provide a standardized format for publishing community definitions. These definitions help applications to interpret the structure and purpose of BGP communities. For example, looking glasses may use the published definitions to parse communities seen in BGP announcements and display their meaning. Another potential use case is in generating routing policy configurations based on community definitions published by an upstream ASN. This could be ~~done~~achieved automatically using external tooling to generate router configurations, or inside a router's command-line interface by importing the definitions and providing the CLI-user with available choices for manual configuration.

~~Note that~~T ~~t~~his document only describes a data model for the publishing format of community definitions. The publishing location and publishing mechanism used are outside the scope of this specification.

Commenté [MB6]: May be mention that this data model does not make an assumption about the protocol used to publish/retrieve formatted data following the model.

4. Tree view

The following tree diagram provides an overview of the "ietf-bgp-communities~~.yang~~" data model.

```
module: draft-ietf-grow-yang-bgp-communities
  +--rw bgp-communities
     +--rw serial          uint32
     +--rw uri?            inet:uri
     +--rw description?    string
     +--rw contact-url?     inet:uri
     +--rw contact~~s~~* [emailaddress]
     |  +--rw email-address           inet:email-address
     |  +--rw name?                   string
     |  +--rw role?                   string
     |  +--rw organization?           string
     |  +--rw organizational-unit?    string
     +--rw regular* [name]
     |  +--rw name            community-name
     |  +--rw category?       community-category
     |  +--rw description?    community-description
     |  +--rw global-admin     two-octet-as-number
     |  +--rw local-admin
     |     +--rw format?   localadmin-format
     |     +--rw field~~s~~* [name]
     |        +--rw name            field-name
     |        +--rw length?         uint8
     |        +--rw pattern         field-pattern
     |        +--rw description?    field-description
     +--rw extended* [name]
     |  +--rw name              community-name
     |  +--rw category?         community-category
     |  +--rw description?      community-description
     |  +--rw type              uint8
     |  +--rw subtype           uint8
```

Commenté [MB7]: According to draft-ietf-netmod-rfc8407bis-22 - Guidelines for Authors and Reviewers of Documents Containing YANG Data Models , «List identifiers SHOULD be singular with the surrounding container name plural. Similarly, "leaf-list" identifiers SHOULD be singular. »

Commenté [MB8]: Easy to read. Feel free to ignore as the base reco is:

«Identifiers SHOULD follow a consistent naming pattern throughout the module. Only lowercase letters, numbers, and dashes SHOULD be used in identifier names. »

Commenté [MB9]: See above

```
|  +--rw (global-admin)
|  |  +--:(asn)
|  |  |  +--rw asn?   two-octet-as-number
|  |  +--:(asn4)
|  |     +--rw asn4?   inet:as-number
|  +--rw local-admin
|     +--rw format?   localadmin-format
|     +--rw fields* [name]
|        +--rw name            field-name
|        +--rw length?         uint8
|        +--rw pattern         field-pattern
|        +--rw description?    field-description
+--rw large* [name]
   +--rw name               community-name
   +--rw category?          community-category
   +--rw description?       community-description
   +--rw global-admin        inet:as-number
   +--rw local-data-part-1
   |  +--rw format?   localadmin-format
   |  +--rw fields* [name]
   |     +--rw name            field-name
   |     +--rw length?         uint8
   |     +--rw pattern         field-pattern
   |     +--rw description?    field-description
   +--rw local-data-part-2
      +--rw format?   localadmin-format
      +--rw fields* [name]
         +--rw name            field-name
         +--rw length?         uint8
         +--rw pattern         field-pattern
         +--rw description?    field-description
```

Figure 1

5. Data ~~elements~~Elements

The BGP Communities YANG ~~Module~~ module contains the elements described in this section. The full contents of the module can be found in Appendix A. Several elements in this module use data types from [I-D.ietf-netmod-rfc6991-bis]. These data types are represented with the prefix "inet".

5.1. The "serial" leaf

A required value of type "uint32", containing the version number for the community set. This value wraps and should be compared using sequence space arithmetic.

5.2. The "uri" leaf

An optional value of type "inet:uri", describing the publication point for the community set.

5.3. The "description" leaf

An optional value of type "string" with a maximum length of 65535, providing information about the specified set of communities.

Commenté [MB10]: Is this really needed given than both forms are covered by as-number

Commenté [MB11]: See above for the guidance in 8407bis

Commenté [MB12]: The module should be included in the main document as it is the main normative piece in this spec

Commenté [MB13]: How is this generated?

5.4. The "contacturl" leaf

An optional value of type "inet:uri", describing a webpage where maintainer contact information may be found.

5.5. The "contacts" list

A list of objects defining contact information for the maintainer(s) of the community set. Each object contains the following elements.

5.5.1. The "emailaddress" leaf

A required value of type "inet:email-address", containing the e-mail address of the contact.

5.5.2. The "name" leaf

An optional value of type "string" with a maximum length of 255, containing the name of the contact.

5.5.3. The "role" leaf

An optional value of type "string" with a maximum length of 255, describing the role of the contact.

5.5.4. The "organization" leaf

An optional value of type "string" with a maximum length of 255, containing the organization of the contact.

5.5.5. The "organizationalunit" leaf

An optional value of type "string" with a maximum length of 255, containing the organizational unit of the contact.

5.6. The "regular" list

A list of objects defining Regular ([RFC1997]) BGP communities. Each object contains the following elements.

5.6.1. The "name" leaf

A required value of type "community-name", containing the name of this community.

The structure of the "community-name" type is defined in Section 6.2.

5.6.2. The "category" leaf

An optional value of type "community-category", containing the category of this community.

The structure of the "community-category" type is defined in Section 6.3.

5.6.3. The "description" leaf

Commenté [MB14]: Consider adding this part to the security cons as displaying random URL has security implication.

May be recommend to render those as txt not clickable links, etc.

An optional value of type "community-description", containing a description of this community.

The structure of the "community-description" type is defined in Section 6.4.

5.6.4. The "globaladmin" leaf

A required value of type "two-octet-as-number", containing the Autonomous ~~Sytem~~ System Number (ASN) set in the Global Administrator part of this community.

The structure of the "two-octet-as-number" type is defined in Section 6.1.

5.6.5. The "localadmin" container

A group of elements that describe the Local Administrator part of the community. This object contains the following elements.

5.6.5.1. The "format" leaf

An optional value of type "localadmin-format", describing the encoding format in which fields are to be parsed (see Section 7.2).

The structure of the "localadmin-format" type is defined in Section 6.5. If this leaf is not defined, the default "decimal" encoding is assumed.

5.6.5.2. The "fields" list

A list of objects that together form the Local Administrator part of the community. The combined length values of all fields MUST NOT exceed the maximum length of the Local Administrator part of the community.

5.6.5.2.1. The "name" leaf

A required value of type "field-name", containing the name of the field.

The structure of the "field-name" type is defined in Section 6.6.

5.6.5.2.2. The "length" leaf

An optional value of type "uint8", containing the length of the field. If the expected field format (Section 5.6.5.1) is "decimal", this is a number of digits. In case the expected field format is "binary", this is a number of bits.

If this leaf is not defined, the length is assumed to be the maximum allowed length of the entire field list. In this case the field list MUST NOT contain more than one element.

Commenté [MB15]: What is the expected use of this?

5.6.5.2.3. The "pattern" leaf

A required value of type "field-pattern", containing a pattern used

for matching the field's contents.

The structure of the "field-pattern" type is defined in Section 6.7.

5.6.5.2.4. The "description" leaf

An optional value of type "field-description", containing a description of the pattern. This description can be used to provide meaning to specific values for a field.

The structure of the "field-description" type is defined in Section 6.8.

5.7. The "extended" list

A list of objects defining Extended ([RFC4360]) BGP communities. Two-Octet and Four-Octet AS Specific communities are supported by this specification. Each object contains the following elements.

5.7.1. The "name" leaf

A required value of type "community-name", containing the name of this community.

The structure of the "community-name" type is defined in Section 6.2.

5.7.2. The "category" leaf

An optional value of type "community-category", containing the category of this community.

The structure of the "community-category" type is defined in Section 6.3.

5.7.3. The "description" leaf

An optional value of type "community-description", containing a description of this community.

The structure of the "community-description" type is defined in Section 6.4.

5.7.4. The "type" leaf

A required value of type "uint8", containing the high-order Type of the community.

5.7.5. The "subtype" leaf

A required value of type "uint8", containing the low-order Sub-Type of the community.

5.7.6. The "asn" leaf

For Two-Octet AS Specific communities: A required value of type "two-octet-as-number", containing the Autonomous Sytem Number set in the Global Administrator part of this community.

Commenté [MB16]: May be to elaborate more.

The structure of the "two-octet-as-number" type is defined in Section 6.1.

5.7.7. The "asn4" leaf

For Four-Octet AS Specific communities: A required value of type "inet:as-number", containing the ~~Autonomous Sytem Number~~ASN set in the Global Administrator part of this community.

5.7.8. The "localadmin" container

A group of elements that describe the Local Administrator part of the community. This object contains the following elements.

5.7.8.1. The "format" leaf

An optional value of type "localadmin-format", describing the encoding format in which fields are to be parsed (see Section 7.2).

The structure of the "localadmin-format" type is defined in Section 6.5. If this leaf is not defined, the default "decimal" encoding is assumed.

5.7.8.2. The "fields" list

A list of objects that together form the Local Administrator part of the community. The combined length values of all fields MUST NOT exceed the maximum length of the Local Administrator part of the community.

The supported leafs in this list are identical to those described in Section 5.6.5.2.

5.8. The "large" list

A list of objects defining Large ([RFC8092]) BGP communities. Each object contains the following elements.

5.8.1. The "name" leaf

A required value of type "community-name", containing the name of this community.
The structure of the "community-name" type is defined in Section 6.2.

5.8.2. The "category" leaf

An optional value of type "community-category", containing the category of this community.

The structure of the "community-category" type is defined in Section 6.3.

5.8.3. The "description" leaf

An optional value of type "community-description", containing a description of this community.

The structure of the "community-description" type is defined in

Section 6.4.

5.8.4. The "globaladmin" leaf

A required value of type "inet:as-number", containing the ~~Autonomous Sytem Number~~ASN set in the Global Administrator part of this community.

5.8.5. The "localdatapart1" container

A group of elements that describe the Local Data Part 1 section of the community. This object contains the following elements.

5.8.5.1. The "format" leaf

An optional value of type "localadmin-format", describing the encoding format in which fields are to be parsed (see Section 7.2).

The structure of the "localadmin-format" type is defined in Section 6.5. If this leaf is not defined, the default "decimal" encoding is assumed.

5.8.5.2. The "fields" list

A list of objects that together form the Local Data Part 1 section of the community. The combined length values of all fields MUST NOT exceed the maximum length of the "Local Data Part 1" section of the community.

The supported leafs in this list are identical to those described in Section 5.6.5.2.

5.8.6. The "localdatapart2" container

A group of elements that describe the Local Data Part 2 section of the community. This object contains the following elements.

5.8.6.1. The "format" leaf

An optional value of type "localadmin-format", describing the encoding format in which fields are to be parsed (see Section 7.2).

The structure of the "localadmin-format" type is defined in Section 6.5. If this leaf is not defined, the default "decimal" encoding is assumed.

5.8.6.2. The "fields" list

A list of objects that together form the Local Data Part 2 section of the community. The combined length values of all fields MUST NOT exceed the maximum length of the "Local Data Part 2" section of the community.

The supported leafs in this list are identical to those described in Section 5.6.5.2.

6. Type definitions

Several of the elements defined in Section 5 use custom data types. These data types are defined here.

6.1. The "two-octet-as-number" data type

A Two-Octet Autonomous System Number, as defined in [RFC1930].

Commenté [MB17]: Not sure this is actually needed

6.2. The "community-name" data type

A string specifying the name of a BGP community. Names may be up to 255 characters long and MUST NOT contain spaces or tabs.

6.3. The "community-category" data type

An enum specifying the category of a BGP community. Possible categories are "informational" and "action", as described in [RFC8195].

6.4. The "community-description" data type

A string specifying the description of a BGP community. Descriptions may be up to 65535 characters long.

6.5. The "localadmin-format" data type

An enum specifying the encoding for a localadmin/localdata field. Possible encodings are "decimal" for decimal numbers and "binary" for bit strings.

6.6. The "field-name" data type

A string specifying the name of a BGP community localadmin/localdata field. Names may be up to 255 characters long and MUST NOT contain spaces or tabs.

6.7. The "field-pattern" data type

A string specifying the pattern of a BGP community localadmin/ localdata field. Patterns may be up to 4095 characters long and are described as POSIX Extended Regular Expressions (see [IEEE.1003-2.1992], ~~section~~ Section 2.8.4).

6.8. The "field-description" data type

A string specifying the description of a BGP community localadmin/ localdata field. Descriptions may be up to 65535 characters long.

7. Operational guidelines

7.1. Publishing guidelines

Operators SHOULD only publish BGP community definitions for networks they control. This may include communities where the Global Administrator field contains a private ASN, if this community has a local meaning inside the network of the publisher.

When publishing community definitions with overlapping field

patterns, these definitions MUST be ordered from most to least preferred. This ensures parsers can perform deterministic matching (see Section 7.2). For example, a definition for a single community "64500:123" needs to be specified before a definition that matches a covering range of communities "64500:*".

7.2. Parsing guidelines

A published BGP community definition can be used by parsers to display information about a received community. If a received community matches multiple published community definitions, the first matching definition in the published order takes precedence.

Parsers that use published community definitions from multiple operators SHOULD NOT attempt to match received communities where the Global Administrator field contains a private ASN, unless they have some method to determine which published definition is the authoritative one.

By default, communities are compared using the decimal representation of the fields. If "format" for a Local Administrator or Local Data Part is set to "binary", the fields in the received community are converted to strings of zeros and ones before comparison.

Applications that parse these community definitions SHOULD reject objects that do not comply with the rules described in this document. Furthermore, parsers SHOULD check that the sum of the specified Local Administrator or Local Data Part field lengths in each community definition does not exceed the local part size of the specified community type. For example, a Regular BGP community definition with format "decimal" containing a field of length 4 and a field of length 2 would be illegal, as the Local Administrator field has a maximum length of 65535 (5 digits).

8. IANA considerations

8.1. YANG Namespace Registration

This document registers the following XML namespace URN in the "IETF XML Registry", following the format defined in [RFC3688]:

URI: urn:ietf:params:xml:ns:yang: ietf-grow-yang-bgp-communities
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
~~TODO~~

8.2. YANG Module Registration

This document registers the following YANG module in the "YANG Module Names" registry [RFC6020]:

Name: ietf-grow-yang-bgp-communities
Maintained by IANA? N
Namespace: urn:ietf:params:xml:ns:yang:ietf-grow-yang-bgp-communities
Prefix: bgp-comm

Reference: RFC XXX~~TODO~~

8.3. YANG SID Allocation

This document registers the following entry in the "IETF YANG SID" registry [I-D.ietf-core-sid]:

TODO

9. Implementation status

RFC-EDITOR: Please remove this section and the accompanying reference(s) before publication.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

9.1. Publishing implementations

The following networks are known to publish BGP community definitions according to this specification.

ASN	Publication URI	YANG model revision
197000	as197000.json (https://www-static.ripe.net/dynamic/draft-ietf-grow-yang-bgp-communities/as197000.json)	2024-06-15
25152	as25152.json (https://www-static.ripe.net/dynamic/draft-ietf-grow-yang-bgp-communities/as25152.json)	2024-06-15

Table 1: Publishing implementations

9.2. Parser implementations

The following known parser implementations exist.

Name	YANG model revision
NLNOG Looking Glass (https://github.com/NLNOG/lg.ring.nlnog.net/)	2024-06-15

Table 2: Parser implementations

10. Security considerations

The YANG module described in this document may be used to specify BGP community definitions in different encoding formats, such as XML, JSON or CBOR. Applications that parse these community definitions SHOULD reject objects that do not comply with the rules described in this document. Furthermore, parsers SHOULD check that the sum of the specified Local Administrator or Local Data Part field lengths in each community definition does not exceed the local part size of the specified community type.

11. Normative References

[I-D.ietf-core-sid]
Veillette, M., Pelov, A., Petrov, I., Bormann, C., and M. Richardson, "YANG Schema Item iDentifier (YANG SID)", Work in Progress, Internet-Draft, draft-ietf-core-sid-24, 22 December 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-core-sid-24>.

[I-D.ietf-netmod-rfc6991-bis]
Schönwälder, J., "Common YANG Data Types", Work in Progress, Internet-Draft, draft-ietf-netmod-rfc6991-bis-15, 23 January 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-rfc6991-bis-15>.

[RFC1930] Hawkinson, J. and T. Bates, "Guidelines for creation, selection, and registration of an Autonomous System (AS)", BCP 6, RFC 1930, DOI 10.17487/RFC1930, March 1996, <https://www.rfc-editor.org/info/rfc1930>.

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

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

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

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

12. Informative References

[IEEE.1003-2.1992] Institute of Electrical and Electronics Engineers, "Information Technology - Portable Operating System Interface (POSIX) - Part 2: Shell and Utilities (Vol. 1)", IEEE Standard 1003.2, IEEE 1003.2-1992, IEEE ieee-1003-2, 1992.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

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

[RFC4384] Meyer, D., "BGP Communities for Data Collection", BCP 114, RFC 4384, DOI 10.17487/RFC4384, February 2006, <https://www.rfc-editor.org/info/rfc4384>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8195] Snijders, J., Heasley, J., and M. Schmidt, "Use of BGP Large Communities", RFC 8195, DOI 10.17487/RFC8195, June 2017, <https://www.rfc-editor.org/info/rfc8195>.

[RFC8792] Watsen, K., Auerswald, E., Farrel, A., and Q. Wu, "Handling Long Lines in Content of Internet-Drafts and RFCs", RFC 8792, DOI 10.17487/RFC8792, June 2020, <https://www.rfc-editor.org/info/rfc8792>.

Appendix A. YANG Module

This section contains the complete YANG module defined in this document.

```
  NOTE: '\' line wrapping per [RFC8792]

module draft-ietf-grow-yang-bgp-communities {

  yang-version 1.1;

  namespace "urn:to-be-defined";

  prefix bgp-comm;

  import ietf-inet-types {
    prefix inet;
    revision-date 2021-02-22;
    reference
      "draft-ietf-netmod-rfc6991-bis-15: Common YANG Data Types";
  }

  organization
    "IETF GROW Working Group";

  contact
    "WG Web:   <https://datatracker.ietf.org/wg/grow/>
     WG List:  <mailto:grow@ietf.org>

     Author:   Martin Pels
       <mailto:mpels@ripe.net>";

  description
    "This module describes a structure for BGP Communities";

  revision "2024-06-15" {
    description
      "Fixed regular expression patterns.";
    reference
      "RFC YYYY: YANG Module for BGP Communities
       RFC-EDITOR: please update YYYY with this RFC ID";
  }

  revision "2024-02-21" {
    description
      "Added data types and string restrictions.
       Added category leaf.";
    reference
      "draft-ietf-grow-yang-bgp-communities-01";
  }

  revision "2023-08-01" {
    description
      "Initial version.";
    reference
      "draft-ietf-grow-yang-bgp-communities-00";
  }

  typedef two-octet-as-number {
    type uint16;
    description
      "This type represents autonomous system numbers, which
```

Commenté [MB18]: It is recommedned to avoid folding for yang. Please check 8407bis

Commenté [MB19]: Please add the boilerplate; refer to the template in 8407bis

Commenté [MB20]: Please check 8407bis

```
     identify an Autonomous System (AS).

     Autonomous system numbers were originally limited to 16
     bits.  BGP extensions have enlarged the autonomous system
     number space to 32 bits.  The two-octet-as-number type uses
     an uint16 base type for use cases where the enlarged number
     space is not supported.";
  reference
    "RFC 1930: Guidelines for creation, selection, and registration
                of an Autonomous System (AS)";
}

typedef community-name {
  type string {
    length 1..255;
    pattern '[^ \t\r\n\p{C}]+';
  }
  description
    "This type restricts values for the name of a BGP community.";
}

typedef community-category {
  type enumeration {
    enum informational {
      value 0;
      description
        "Informational community";
    }
    enum action {
      value 1;
      description
        "Action community";
    }
  }
  description
    "This type restricts values for the category of a BGP community.";
}

typedef community-description {
  type string {
    length 1..65535;
    pattern '[^\p{C}]+';
  }
  description
    "This type restricts values for the description of a BGP
     community.";
}

typedef localadmin-format {
  type enumeration {
    enum decimal {
      value 0;
      description
        "Decimal number string";
    }
    enum binary {
      value 1;
      description
```

```
        "Bit string";
      }
    }
    description
      "This type defines the format options for a BGP community
       localadmin/localdata field encoding";
  }

  typedef field-name {
    type string {
      length 1..255;
      pattern '[^ \t\r\n\p{C}]+';
    }
    description
      "This type restricts values for the name leaf of a BGP community
       localadmin/localdata field.";
  }

  typedef field-pattern {
    type string {
      length 1..4095;
      pattern '[-0-9.,*?^$+|(){}\[\]]+';
    }
    description
      "This type restricts values for the pattern leaf of a BGP
       community localadmin/localdata field.  Patterns are
       described as POSIX Extended Regular Expressions";
    reference
      "IEEE 1003.2-1992: Information Technology - Portable
       Operating System Interface (POSIX) - Part 2: Shell and
       Utilities (Vol. 1)";
  }

  typedef field-description {
    type string {
      length 1..65535;
      pattern '[^\p{C}]+';
    }
    description
      "This type restricts values for the description leaf of a BGP
       community localadmin/localdata field.";
  }

  grouping local-admin-fields {
    list fields {
      ordered-by user;
      key "name";

      leaf name {
        type field-name;
        mandatory true;
        description
          "The name of the field";
      }

      leaf length {
        type uint8;
        description
```

Commenté [MB21]: To ease readability of the module

Commenté [MB22]: As this is a key

```
          "Length of the field";
      }

      leaf pattern {
        type field-pattern;
        mandatory true;
        description
          "Regular Expression describing the expected contents of
           the field";
      }

      leaf description {
        type field-description;
        description
          "A text description of the field contents";
      }
    }
  }

  container bgp-communities {

    leaf serial {
      type uint32;
      mandatory true;
      description
        "Version number of the community set";
    }

    leaf uri {
      type inet:uri;
      description
        "Publication point for the community set";
    }

    leaf description {
      type string {
        length 1..65535;
        pattern '[^\p{C}]+';
      }
      description
        "A description for the community set";
    }

    leaf contact-url {
      type inet:uri;
      description
        "A reference to a webpage with maintainer contact information";
    }

    list contacts {
      key "emailaddress";

      leaf emailaddress {
        type inet:email-address;
        description
          "Maintainer contact e-mail address";
      }
```

Commenté [MB23]: Not sure how this useful here

```
      leaf name {
        type string {
          length 1..255;
          pattern '[^\p{C}]+';
        }
        description
          "Maintainer contact name";
      }

      leaf role {
        type string {
          length 1..255;
          pattern '[^\p{C}]+';
        }
        description
          "Maintainer contact role";
      }

      leaf organization {
        type string {
          length 1..255;
          pattern '[^\p{C}]+';
        }
        description
          "Maintainer contact organization";
      }

      leaf organizational-unit {
        type string {
          length 1..255;
          pattern '[^\p{C}]+';
        }
        description
          "Maintainer contact organizational unit";
      }
    }

    list regular {
      key "name";

      leaf name {
        type community-name;
        mandatory true;
        description
          "Community name";
      }

      leaf category {
        type community-category;
        description
          "Category of the community";
      }

      leaf description {
        type community-description;
        description
          "Description for the community";
      }
```

```
        leaf global-admin {
          type two-octet-as-number;
          mandatory true;
          description
            "Global Administrator field";
        }

        container local-admin {
          leaf format {
            type localadmin-format;
            default decimal;
            description
              "Format used for parsing localadmin fields";
          }

          uses localadmin-fields;
        }

        description
          "A list of objects defining Regular BGP Communities";
        reference
          "RFC_1997: BGP Communities Attribute";
      }

      list extended {
        key "name";

        leaf name {
          type community-name;
          mandatory true;
          description
            "Community name";
        }

        leaf category {
          type community-category;
          description
            "Category of the community";
        }

        leaf description {
          type community-description;
          description
            "Description for the community";
        }

        leaf type {
          type uint8;
          mandatory true;
          description
            "Type Field";
        }
        leaf subtype {
          type uint8;
          mandatory true;
          description
            "Sub-Type Field";
```

```
      }

      choice global-admin {
        mandatory true;
        case asn {
          leaf asn {
            type two-octet-as-number;
            description
              "Two-Octet AS";
          }
        }
        case asn4 {
          leaf asn4 {
            type inet:as-number;
            description
              "Four-Octet AS";
          }
        }
      }

      container local-admin {
        leaf format {
          type localadmin-format;
          default decimal;
          description
            "Format used for parsing localadmin fields";
        }

        uses localadmin-fields;
      }

      description
        "A list of objects defining Extended BGP Communities";
      reference
        "RFC_4360: BGP Extended Communities Attribute";
    }

    list large {
      key "name";

      leaf name {
        type community-name;
        mandatory true;
        description
          "Community name";
      }

      leaf category {
        type community-category;
        description
          "Category of the community";
      }

      leaf description {
        type community-description;
        description
          "Description for the community";
      }
```

```
        leaf global-admin {
          type inet:as-number;
          mandatory true;
          description
            "Global Administrator field";
        }

        container localdatapart1 {
          leaf format {
            type localadmin-format;
            default decimal;
            description
              "Format used for parsing localadmin fields";
          }

          uses local-admin-fields;
        }

        container localdatapart2 {
          leaf format {
            type localadmin-format;
            default decimal;
            description
              "Format used for parsing localadmin fields";
          }

          uses localadmin-fields;
        }

        description
          "A list of objects defining Large BGP Communities";
        reference
          "RFC_8092: BGP Large Communities Attribute";
      }
    }
}
```

Commenté [MB24]: Same description. Consider adding a meaningful description

~~Figure 2~~

Appendix B. JSON Examples

This section shows example use cases for the YANG module defined in this document, using JSON[RFC7951] encoding.

B.1. RFC8195 Selective NO_EXPORT definition

A JSON definition for the example Large BGP community described in [RFC8195], section 4.1.1 looks as follows.

```
{
  "draft-ietf-grow-yang-bgp-communities:bgp-communities": {
    "serial": 2023080101,
    "uri": "http://example.net/peering/communities",
    "description": "BGP Community example for ASN-Based Selective \
NO_EXPORT",
    "contacts": [
```

```
        {
          "emailaddress": "noc@example.net",
          "name": "Example.net contact",
          "role": "Administrative contact",
          "organization": "Example.net",
          "organizationalunit": "NOC"
        }
      ],
      "large": [
        {
          "name": "RFC8195-NOEXPORT-ASN",
          "category": "action",
          "description": "Do not export route to ASN",
          "globaladmin": 65539,
          "localdatapart1": {
            "fields": [
              {
                "name": "Function",
                "pattern": "4",
                "description": "ASN-No-Export"
              }
            ]
          },
          "localdatapart2": {
            "fields": [
              {
                "name": "ASN",
                "pattern": ".*"
              }
            ]
          }
        }
      ]
    }
  }
```

Figure 3

B.2. RFC4384 Data Collection definition

A JSON definition for the example Regular BGP community described in [RFC4384], section 4 looks as follows.

NOTE: '\' line wrapping per [RFC8792]

```
 {
   "draft-ietf-grow-yang-bgp-communities:bgp-communities": {
     "serial": 2023080101,
     "uri": "http://example.net/peering/communities",
     "description": "BGP Community example for Data Collection",
     "contacturl": "https://example.net/contact",
     "regular": [
       {
         "name": "RFC4384-ORIGIN-OC/FJ",
         "description": "A national route over a terrestrial link from \
 the Fiji Islands",
         "globaladmin": 64497,
```

```
        "localadmin": {
          "format": "binary",
          "fields": [
            {
              "name": "Region",
              "length": 5,
              "pattern": "00010",
              "description": "OC"
            },
            {
              "name": "Satellite",
              "length": 1,
              "pattern": "0"
            },
            {
              "name": "Country",
              "length": 10,
              "pattern": "0011110010",
              "description": "FJ"
            }
          ]
        }
      }
    ]
  }
}
```

Figure 4

Appendix C. Acknowledgements

The author would like to thank Jeffrey Haas, Luuk Hendriks, Jasper den Hertog, Teun Vink, Tom Petch and Dale Carder for contributing ideas and feedback to this document.

Author's Address

Martin Pels
RIPE NCC
Netherlands
Email: mpels@ripe.net