

IDR Working Group
INTERNET-DRAFT
Intended status: Experimental
Expires: Feb 23, 2022

Louis Chan
Juniper Networks
Aug 23, 2021

Color Operation with BGP Label Unicast
draft-chan-idr-bgp-lu2-04.txt

Abstract

This document specifies how to carry colored path advertisement via an enhancement to the existing protocol BGP Label Unicast. It would allow backward compatibility with RFC8277.

The targeted solution is to use stack of labels advertised via BGP Label Unicast 2.0 for end to end traffic steering across multiple IGP domains. The operation is similar to Segment Routing.

This proposed protocol will convey the necessary reachability information to the ingress PE node to construct an end to end path.

Another two problems addressed here are the interworking with Flex-Algo, and the MPLS label space limit problem.

Please note that there is a major change of protocol format starting from version 01 draft. Except the optional BGP capability code, these rest of BGP attributes used in this draft are defined in previous RFC or in use today in other scenarios.

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 Feb 23, 2022.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors.

All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction.....	2
2. Conventions used in this document.....	4
3. Carrying Label Mapping Information with Color and Label Stack..	4
3.1. Use of Add-path to advertise multiple color paths.....	4
3.2. Color extended community for BGP Labeled Unicast.....	5
3.3. Color extended community for service prefixes.....	6
3.4. Color Slicing Capability.....	6
4. Uniqueness of path entries.....	7
5. AIGP consideration.....	8
6. Explicit Withdraw of a <path-id, color(s), prefix>.....	8
7. Error Handling Procedure.....	8
8. Controller Compatibility.....	8
9. Interworking with Flex Algo.....	9
10. Label stacking to increase label space.....	9
11. Tunneling SRv6 packet via MPLS.....	9
12. Security Considerations.....	10
13. IANA Considerations.....	10
14. References.....	10
14.1. Normative References.....	10
14.2. Informative References.....	10
15. Acknowledgments.....	11

1. Introduction

The proposed protocol is aimed to solve interdomain traffic steering, with different transport services in mind. One application is low latency service across multiple IGP domains, which could scale up to 100k or more routers network.

BGP is a flexible protocol. With additional of color attribute to BGP Label Unicast, a path with specific color would be given a meaning in application - a low latency path, a fully protected path, or a path for diversity.

The stack of labels would mean an end to end path across domains through each ABR or ASBR. Each ABR or ASBR will take one label from the stack, and hence pick the forwarding path to next ABR, ASBR, or the final destination.

And the label in the stack may be derived from any of the below

Chan

Expires Feb 23, 2022

[Page 2]

- Prefix SID
- Binding SID for RSVP LSP
- Binding SID for SR-TE LSP
- Local assigned label

The enhancement to the original RFC8277 is to add color extended community, with multiple advertisement allowed. The result is similar to multi-topology BGP-LU with different colors.

With Add-path [RFC7911] feature, non color RIB and colored RIB could be advertised to the BGP neighbors without new additional attributes. Add-path capability is required advertise multiple paths with same prefix but different colors.

A new [BGP-CAP] should be required to enforce such slicing operation during negotiation.

On the other hand, to enable the service prefixes to be mapped accordingly, the L3VPN, L2VPN, EVPN and IP prefix with BGP signaling, the color extended community is also added there. In the PE node, the service prefixes with color will be matched to a transport tunnel with the same color.

The following is an example. Between PE1 and PE2, there is a VPN service running with label 16, which is associated with color 100.

PE1-----ABR1-----ABR2-----PE2

PE1 will send the following labels with a color 100 path plus VPN label

[2001 13001 801 16], where

2001 - SR label to reach ABR1

13001 - a Binding-SID label for ABR1-ABR2 tunnel. Underlying tunnel type is RSVP-TE

801 - a Binding-SID label for ABR2-PE2 tunnel. Underlying tunnel type is SR-TE

16 - a VPN label, which is signaled via other means

[2001 13001 801] - denotes the label stack for this color 100 path to reach PE2

The document here is going to describe how PE1 gains enough information to build this label stack across routing domains.

If PE1 wants to reach PE2 with another colored path, say color 200, the label stack could be different.

At the same time, this architecture is also controller friendly, since all the notation is Segment Routing compatible, like use of Binding-SID.

The above architecture could be used in conjunction with Flex-Algo [FLEXAGLO]
where
one color could represent a Flex Algorithm. e.g. color 128 equals to Algo 128

When using with Flex Algo in huge network, there could be label space limit. The MPLS label 20 bits long and the maximum label space is around 1 million. In order to represent more IPv4 or IPv6 nodes, label stacking method is recommended. On the IP loopback address could be represent by one or more labels. In this case, (20 bits x n) of label address space is possible.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying significance described in RFC 2119.

3. Carrying Label Mapping Information with Color and Label Stack

3.1. Use of Add-path to advertise multiple color paths

The use of Path Identifier is to allow multiple advertisement of the same prefix but with different colors or null color.

The extended NLRI format would be like this

```
+-----+
| Path Identifier (4 octets) |
+-----+
| Length (1 octet)         |
+-----+
| Label (3 octets)         | ~
+-----+
~ Label (3 octets)         |
+-----+
| Prefix (variable)        |
+-----+
```


3.2. Color extended community for BGP Labeled Unicast

The addition of Color Extended Community is an opaque extended community from RFC4360 and RFC5512. The draft allows multiple color values advertisement.

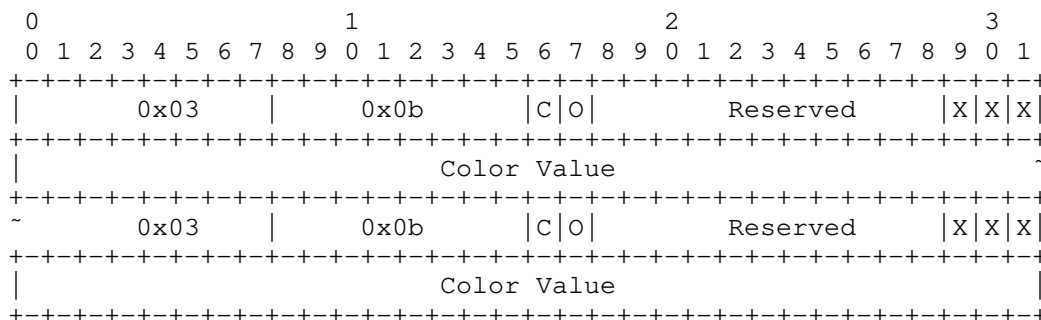


Figure 1: Color value advertisement format

Both in BGP update and MP_UNREACH_NLRI message, multiple color extended communities could be included. It means that multiple colors, indicating different kind of services, could share the same label stack. With the use of Path-ID, the multiple colors are considered as one bundled update. Any subsequent update is based on Path-ID.

If color extended community is not present in a BGP update message, it would be treated as normal BGP-LU without any color.

3 bits of XXX is reserved here for the draft.

The meaning for XXX is interpreted as sub-slice of color, with 0 to 7 in decimal, or 000b and 111b in binary. These sub-slice could be used in either of the following case.

- a) Primary path and fallback paths in order of preference
 - 0 - primary path
 - 1 - first and most preferred backup path
 -
 - 7 - least preferred backup path

- b) ECMP paths up to 8, since all paths should be active in forwarding plane.

Color value 0 is reserved for future interoperability purpose.

Color value 1 - 31 are not recommended to use, and this range is reserved for future use.

3.3. Color extended community for service prefixes

The same format of color extended community is advertised with service prefixes, which could be VPN prefixes or IP prefixes. The order of the color extended community could be interpreted as

- Order of primary and fallback colors
- Or, ECMP of equal split between color paths

The above would be interpreted by the receiving PE upon its local configuration.

It is optional to enable sub-slice notation.

But if sub-slice bits are used, it will be used to map directly to each of the sub-slice path. If sub-slice path is not available for mapping, it should just fallback to resolving by color.

3.4. Color Slicing Capability

The Color Slicing Capability is a BGP capability [RFC5492], with Capability Code (TBD).

The color slicing capability is an optional but preferred to have capability. It could be configurable parameters at both side of BGP session but with assumption of BGP add-path support [RFC7911]. If the specific BGP capability is not negotiated, it is assumed version 0 without sub-slice notation. In this case, multiple paths with color attribute are advertised through BGP add-path.

The Capability Length field of this capability is variable. The Capability Value field consists of one or more of the following tuples:

Address Family Identifier (2 octets)
Subsequent Address Family Identifier (1 octet)
version (1 octet)
Reserved (3 octet)

Chan

Expires Feb 23, 2022

[Page 6]

The meaning and use of the fields are as follows:

Address Family Identifier (AFI):

This field is the same as the one used in [RFC4760].

Subsequent Address Family Identifier (SAFI):

This field is the same as the one used in [RFC4760].

Version:

This field is for capability negotiation.

```

  0 1 2 3 4 5 6 7
+--+--+--+--+--+--+
|v v v v|      |s|
+--+--+--+--+--+--+
```

Each of 4 bits of v represents a flag of version from 0 to 4, where LSB denotes support of version 1, and MSB denotes version 4. Version 0 is the default mode of operation, which is described in this document. To determine the common capability between the two BGP PEER, logical AND function to use determine the highest denominator of protocol version.

For example, if BGP receive 0b0110 from its peer and perform AND function with its own capability 0b0010, the result is 0b0010. Version 2 is selected.

The other examples are

- 0b0110 AND 0b0110, version 3 is selected
- 0b0100 AND 0b0010, version 0 is selected

Version 1 (0b0001) is reserved.

S-flag is the indication of use of sub-slice. Set to 1 if sub-slice notation is enforced. If either side is set to 0 for S-flag, sub-slice is not in use.

Reserved:

This field is reserved for future use.

4. Uniqueness of path entries

a) Use of color can be considered to slice into multiple BGP Label Unicast RIB. Therefore, it should be treated as unique entries for the <path-id, color(s), prefix>.

e.g. <path-id, color(s), prefix>, [labels]

<123, 100, 10.1.1.1/32>, [1000 2000]

<124, 200, 10.1.1.1/32>, [1000 2000]

Chan

Expires Feb 23, 2022

[Page 7]

<222, {300,400}, 10.1.1.1/32>, [1000 2000]

<223, null, 10.1.1.1/32>, [1000 2000]

All these 4 NLRI are considered different but valid entries for different color instances.

b) With sub-slice notation

<path-id, color-sub, prefix>, [labels]

<901, 100-0, 10.1.1.1/32>, [1000 2000]

<902, 100-1, 10.1.1.1/32>, [1001 3000]

<903, 100-7, 10.1.1.1/32>, [1002 4000]

These 3 NLRI are distinct, and the second and third NLRI could be used for backup or ECMP purpose.

5. AIGP consideration

AIGP (RFC7311) would be also used in here to embed certain metric across.

6. Explicit Withdraw of a <path-id, color(s), prefix>

According to RFC8277, MP_UNREACH_NLRI can be used to remove binding of a <path-id, color(s), prefix>.

If a path-id is associated with a prefix with multiple colors, the withdrawal would be applied to all associated colors.

To withdraw color(s) partially from the same path-id advertisement, BGP update should be used instead.

7. Error Handling Procedure

If BGP receiver could not handle the NLRI, it should silently discard with error logging.

8. Controller Compatibility

The proposed architecture is compatible with controller for end to end provisioning. Persistent label, like Binding-SID is recommended to be used. Hence, controller could learn these labels from the network, and program specific end to end path.

In this case, BGP-LU2 will provide a second best path to an ingress PE node, while a controller, with more external information, could provide a best path from overall perspective.

Controller could also be deployed based on domain by domain perspective. e.g. Optimizing latency of a RSVP LSP, or maintain the bandwidth and loading between SR-TE LSPs.

9. Interworking with Flex Algo

Flex Algo is a way of network slicing, but it is only an IGP protocol. In order to scale across different domains, BGP is recommended as the method to distribute the information across.

With color notation in this proposal, one router can distribute to another domain via BGP.

There are two ways of mapping Flex-Algo to color attribute in BGP-LU2

- a) Color 128 equals Flex Algo 128
- b) Or, Color 400 is mapped to Flex Algo 128

10. Label stacking to increase label space

Due to the use of Flex-Algo [FLEXALGO], the MPLS label space might run into limit. Each node will need extra labels for each Algo.

The idea is to use multiple labels to represent a single node. In this case, the label space becomes $(2^{20})^n$, depending on n stacking level.

For IPv6 address, there would be enough label space even if running with SR-MPLS.

For example, for node 1.1.1.1, 2 consecutive labels are used to represent the node.

Algo 0: [100101 1000001]

Algo 128: [400101 4000001]

How the forwarding plane treats the stacked labels is out of the discussion here.

11. Tunneling SRv6 packet via MPLS

PE1-----ABR1-----ABR2-----PE2

In a SRv6 network, PE1 and PE2 is using SRv6 for VPN service. Between ABR1 and ABR2, it is capable of MPLS only. The use of BGP-LU2 would be a method to prov

ide

locator route mapping to MPLS tunnel between ABRs.

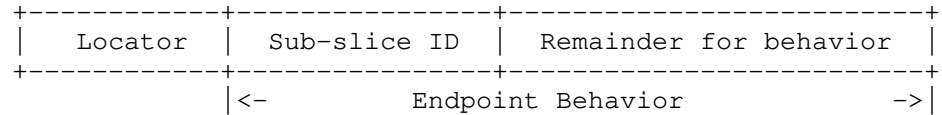
At ABR1, the mapping options could be

Chan

Expires Feb 23, 2022

[Page 9]

- a) Use of color attribute associated with the VPN advertisement and map to the desired tunnel.
- b) Up to the locator route. For example, use first 48 bits of SRv6 header FC00:0000:nnnn::/48 ; where nnnn is the locator portion
- c) Making use of sub-slice information as defined in [SRV6-SUBSLICE]



Sub-slice ID could be used for mapping to different color path in MPLS. For example,

FC00:0000:nnnn:ssss::/64 ; where ssss is a sub-slice ID

ABR2 advertises a/64 prefix route inclusive of sub-slice ID via BGP-LU2 into ABR1. Hence, traffic will be redirected to a MPLS tunnel from ABR1.

- d) With the format described in [SRV6-SUBSLICE], a mapping could be made between sub-slice ID and <color, sub-slice> mentioned in section 3.2.

12. Security Considerations

TBD

13. IANA Considerations

TBD. It will require a new BGP capability code to enable such color operation.

New SAFI might be required as well.

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels",
BCP 14, RFC 2119, March 1997.

14.2. Informative References

[RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, DOI 10.17487/RFC3107, May 2001,

<<https://www.rfc-editor.org/info/rfc3107>>.

[RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, February 2006

<<https://www.rfc-editor.org/info/rfc4360>>.

[RFC5512] Mohapatra, P. and E. Rosen, "The BGP Encapsulation Subsequent Address Family Identifier (SAFI) and the BGP Tunnel Encapsulation Attribute", RFC 5512, April 2009.

<<https://www.rfc-editor.org/info/rfc5512>>.

[RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J., and D. McPherson, "Dissemination of Flow Specification Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009,

<<http://www.rfc-editor.org/info/rfc5575>>.

[RFC7311] Mohapatra, P., Fernando, R., Rosen, E., and J. Uttaro, "The Accumulated IGP Metric Attribute for BGP", RFC 7311, DOI 10.17487/RFC7311, August 2014,

<<https://www.rfc-editor.org/info/rfc7311>>.

[RFC7911] Walton, D., Retana, A., Chen, E., and J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, DOI 10.17487/RFC7911, July 2016,

<<https://www.rfc-editor.org/info/rfc7911>>.

[RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017,

<<https://www.rfc-editor.org/info/rfc8277>>.

[BGP-CAP] Chandra, R. and J. Scudder, "Capabilities Advertisement with BGP-4", RFC 2842, May 2000.

[FLEXAGLO] S. Hegde, P. Psenak and etc, IGP Flexible Algorithm
<https://datatracker.ietf.org/doc/draft-ietf-lsr-flex-algo>

[SRV6-SUBSLICE] Louis Chan, Sub-slicing for SRv6
<https://datatracker.ietf.org/doc/draft-chan-srv6-sub-slice/>

15. Acknowledgments

The following people have contributed to this document:

Jeff Haas, Juniper Networks

Shraddha Hedge, Juniper Networks

Chan

Expires Feb 23, 2022

[Page 11]

Santosh Kolenchery, Juniper Networks
Shihari Sangli, Juniper Networks
Krzysztof Szarkowicz, Juniper Networks
Yimin Shen, Juniper Networks

Author Address

Louis Chan (editor)
Juniper Networks
2604, Cityplaza One, 1111 King's Road
Taikoo Shing
Hong Kong

Phone: +85225876659
Email: louis@juniper.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 20 October 2022

H. Chen
Futurewei
M. Toy
Verizon
A. Wang
China Telecom
Z. Li
China Mobile
L. Liu
Fujitsu
X. Liu
Volta Networks
18 April 2022

SR Path Ingress Protection
draft-chen-idr-sr-ingress-protection-06

Abstract

This document describes extensions to Border Gateway Protocol (BGP) for protecting the ingress node of a Segment Routing (SR) tunnel or path.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 October 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Terminologies	3
3. SR Path Ingress Protection Example	4
4. Behavior after Ingress Failure	4
5. Extensions to BGP	5
5.1. SR Path Ingress Protection Sub-TLV	5
5.1.1. Primary Ingress Sub-TLV	6
5.1.2. Service Sub-TLV	7
5.1.3. Traffic Description Sub-TLVs	8
6. Backup Ingress Behavior	11
7. Security Considerations	12
8. Acknowledgements	13
9. IANA Considerations	13
9.1. BGP Tunnel Encapsulation Attribute Sub-TLVs	13
9.2. Ingress Protection Information Sub-TLVs	13
10. References	13
10.1. Normative References	13
10.2. Informative References	14
Authors' Addresses	15

1. Introduction

The fast protection of a transit node of a Segment Routing (SR) path or tunnel is described in [I-D.bashandy-rtgwg-segment-routing-ti-lfa] and [I-D.hu-spring-segment-routing-proxy-forwarding]. [RFC8424] presents extensions to RSVP-TE for the fast protection of the ingress node of a traffic engineering (TE) Label Switching Path (LSP). However, these documents do not discuss any protocol extensions for the fast protection of the ingress node of an SR path or tunnel.

This document fills that void and specifies protocol extensions to Border Gateway Protocol (BGP) for the fast protection of the ingress node of an SR path or tunnel. Ingress node and ingress, fast protection and protection as well as SR path and SR tunnel will be used exchangeably in the following sections.

2. Terminologies

The following terminologies are used in this document.

SR: Segment Routing

SRv6: SR for IPv6

SRH: Segment Routing Header

SID: Segment Identifier

CE: Customer Edge

PE: Provider Edge

LFA: Loop-Free Alternate

TI-LFA: Topology Independent LFA

TE: Traffic Engineering

BFD: Bidirectional Forwarding Detection

VPN: Virtual Private Network

L3VPN: Layer 3 VPN

FIB: Forwarding Information Base

PLR: Point of Local Repair

BGP: Border Gateway Protocol

IGP: Interior Gateway Protocol

OSPF: Open Shortest Path First

IS-IS: Intermediate System to Intermediate System

3. SR Path Ingress Protection Example

To protect against the failure of the (primary) ingress node of a (primary) SR path, a backup ingress node is configured or selected and is different from the (primary) ingress node. A backup SR path from the backup ingress node is computed and installed. Primary ingress and ingress as well as primary SR path and SR path will be used exchangeably.

Figure 1 shows an example of protecting ingress PE1 of a SR path, which is from ingress PE1 to egress PE3.

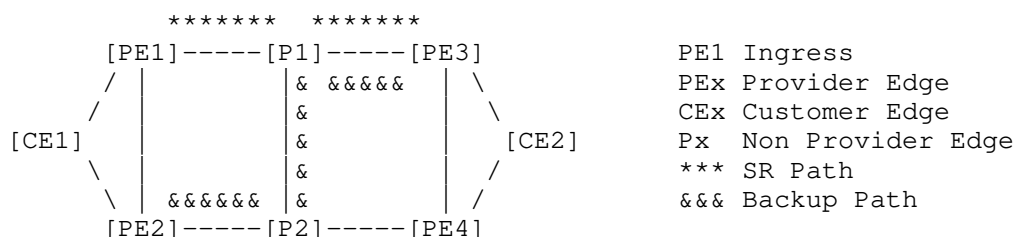


Figure 1: Protecting Ingress PE1 of SR Path PE1-P1-PE3

In normal operations, CE1 sends the traffic with destination PE3 to ingress PE1, which imports the traffic into the SR path.

When CE1 detects the failure of ingress PE1, it switches the traffic to backup ingress PE2, which imports the traffic from CE1 into a backup SR path. The backup path is from the backup ingress PE2 to the egress PE3. When the traffic is imported into the backup path, it is sent to the egress PE3 along the path.

4. Behavior after Ingress Failure

After the failure of the ingress of an SR path happens, there are a couple of different ways to detect the failure. In each way, there may be some specific behavior for the traffic source (e.g., CE1) and the backup ingress (e.g., PE2).

In one way, the traffic source (e.g., CE1) is responsible for fast detecting the failure of the ingress (e.g., PE1) of an SR path. Fast detecting the failure means detecting the failure in a few or tens of milliseconds. The backup ingress (e.g., PE2) is ready to import the traffic from the traffic source into the backup SR path installed.

In normal operations, the source sends the traffic to the ingress of the SR path. When the source detects the failure of the ingress, it switches the traffic to the backup ingress, which delivers the traffic to the egress of the SR path via the backup SR path.

In another way, the backup ingress is responsible for fast detecting the failure of the ingress of an SR path.

In normal operations, the source (e.g., CE1) sends the traffic to the ingress (e.g., PE1) and may send the traffic to the backup ingress (e.g., PE2). It sends the traffic to the backup ingress (e.g., PE2) after the ingress fails.

The backup ingress does not import any traffic from the source into the backup SR path in normal operations. When it detects the failure of the ingress, it imports the traffic from the source into the backup SR path.

5. Extensions to BGP

For a SR path from a primary ingress node to an egress node, a backup ingress node is selected to protect the failure of the primary ingress node of the SR path. This section describes the extensions to BGP for representing the information for protecting the primary ingress node in a BGP UPDATE message and distributing the information to the backup ingress node. The information includes a SR backup path.

[I-D.ietf-idr-segment-routing-te-policy] specifies a way of representing a SR path in a BGP UPDATE message and distributing the SR path to the ingress node of the SR path.

This is extended to represent the information for protecting the primary ingress by defining a few of new Sub-TLVs.

5.1. SR Path Ingress Protection Sub-TLV

A new Sub-TLV, called SR Path Ingress Protection Sub-TLV, is defined. When a UPDATE message is sent to the backup ingress node for protecting the primary ingress node of a SR path, the message contains this Sub-TLV. Its format is illustrated below.

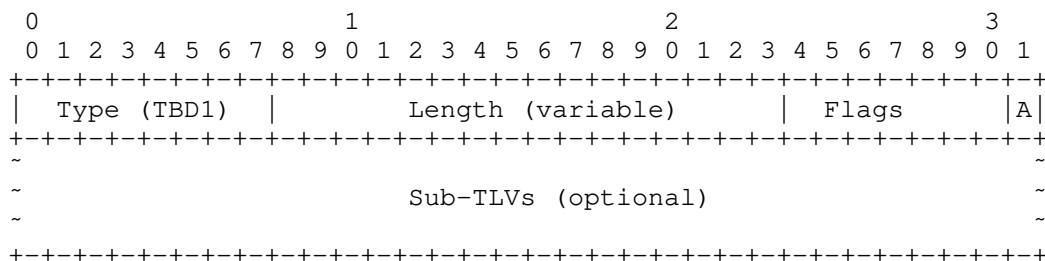


Figure 2: SR Path Ingress Protection Sub-TLV

Type: TBD1 is to be assigned by IANA.

Length: Variable.

Flags: 1 octet. One flag is defined.

Flag A: 1 bit. It is set to

- 1: request a backup ingress to let the forwarding entry for the backup SR path be Active.
- 0: request a backup ingress to let the forwarding entry for the backup SR path be inactive initially and to make the entry be active after detecting the failure of the primary ingress node of the primary SR path.

A few optional Sub-TLVs are defined, which are Primary Ingress Sub-TLV, Service Sub-TLV and Traffic Description Sub-TLV.

5.1.1. Primary Ingress Sub-TLV

A Primary Ingress Sub-TLV indicates the IP address of the primary ingress node of a primary SR path. It has two formats: one for primary ingress node IPv4 address and the other for primary ingress node IPv6 address, which are illustrated below.

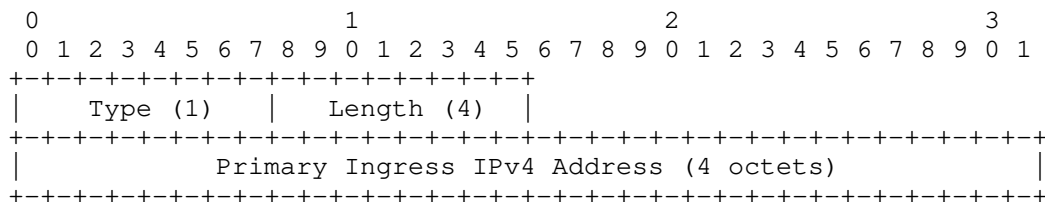


Figure 3: Primary Ingress IPv4 Address Sub-TLV

Type: Its value (1 suggested) is to be assigned by IANA.

Length: 4.

Primary Ingress IPv4 Address: 4 octets. It represents an IPv4 host address of the primary ingress node of a primary SR path.

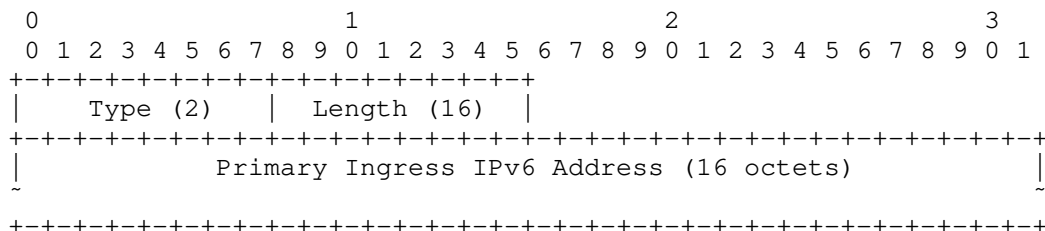


Figure 4: Primary Ingress IPv6 Address Sub-TLV

Type: Its value (2 suggested) is to be assigned by IANA.

Length: 16.

Primary Ingress IPv6 Address: 16 octets. It represents an IPv6 host address of the primary ingress node of a primary SR path.

5.1.2. Service Sub-TLV

A Service Sub-TLV contains a service ID or label to be added into a packet to be carried by a SR path. It has three formats: the first one for the service identified by a label, the second one for the service identified by a service identifier (ID) of 32 bits, and the third one for the service identified by a service identifier (ID) of 128 bits. Their formats are illustrated below.

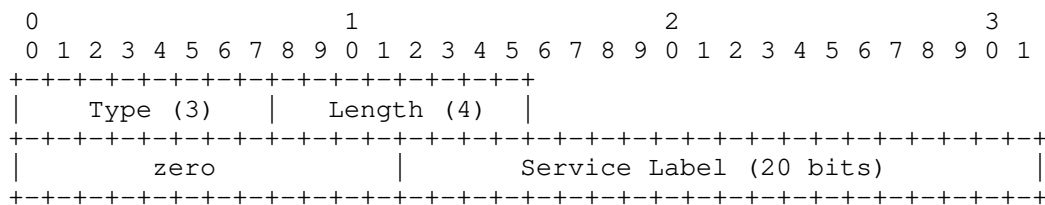


Figure 5: Service Label Sub-TLV

Type: Its value (3 suggested) is to be assigned by IANA.

Length: 4.

Service Label: the least significant 20 bits. It represents a label of 20 bits.

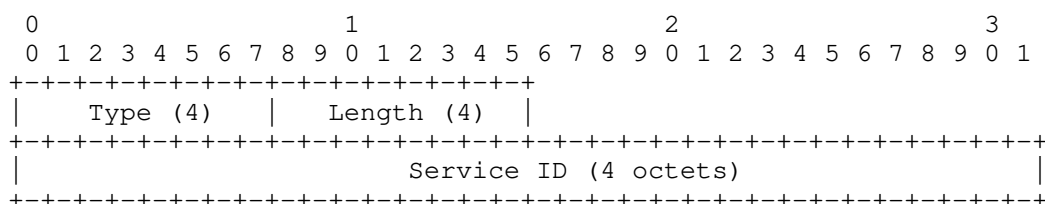


Figure 6: 32 Bits Service ID Sub-TLV

Type: Its value (4 suggested) is to be assigned by IANA.

Length: 4.

Service ID: 4 octets. It represents a Service Identifier (ID) of 32 bits.

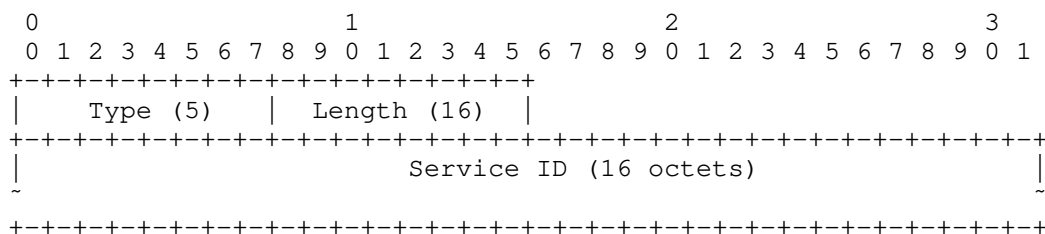


Figure 7: 128 Bits Service ID Sub-TLV

Type: Its value (5 suggested) is to be assigned by IANA.

Length: 16.

Service ID: 16 octets. It represents a Service Identifier (ID) of 128 bits.

5.1.3. Traffic Description Sub-TLVs

A Traffic Description Sub-TLV describes the traffic to be imported into a backup SR path. Five Traffic Description Sub-TLVs are defined. Two of them are FEC Sub-TLVs and the others are interface Sub-TLVs.

Two FEC Sub-TLVs are IPv4 and IPv6 FEC Sub-TLVs. Their formats are illustrated below.

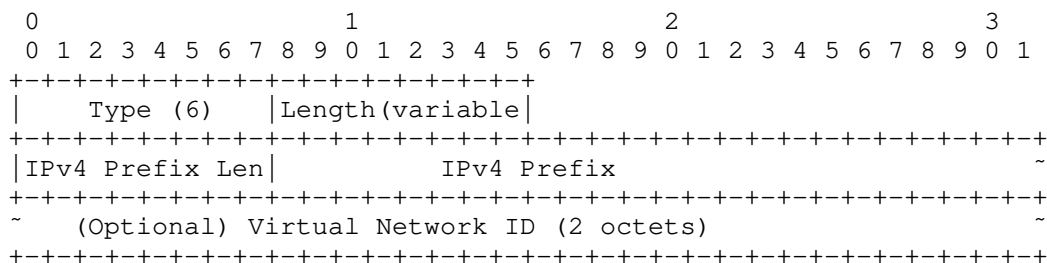


Figure 8: IPv4 FEC Sub-TLV

Type: Its value (6 suggested) is to be assigned by IANA.

Length: Variable.

IPv4 Prefix Len: Indicates the length of the IPv4 Prefix.

IPv4 Prefix: IPv4 Prefix rounded to octets.

Virtual Network ID: 2 octets. This is optional. It indicates the ID of a virtual network.

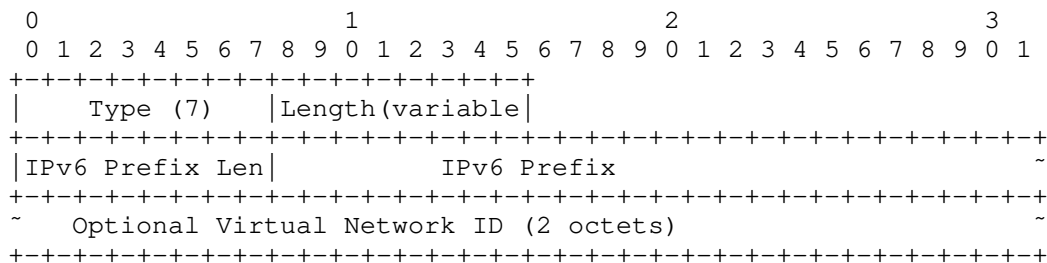


Figure 9: IPv6 FEC Sub-TLV

Type: Its value (7 suggested) is to be assigned by IANA.

Length: Variable.

IPv6 Prefix Len: Indicates the length of the IPv6 Prefix.

IPv6 Prefix: IPv6 Prefix rounded to octets.

Virtual Network ID: 2 octets. This is optional. It indicates the ID of a virtual network.

An Interface sub-TLV indicates the interface from which the traffic is received and imported into the backup SR path/tunnel. It has three formats: one for interface index, the other two for IPv4 and IPv6 address, which are illustrated below.

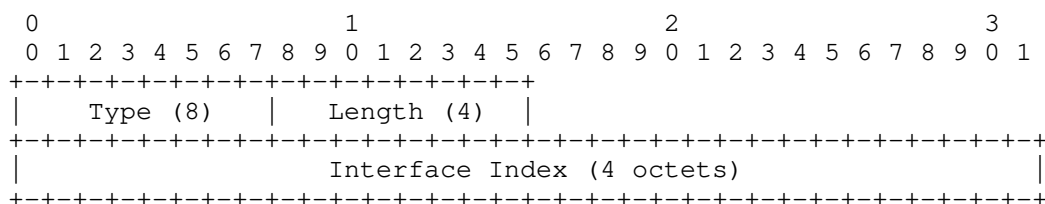


Figure 10: Interface Index Sub-TLV

Type: Its value (8 suggested) is to be assigned by IANA.

Length: 4.

Interface Index: 4 octets. It indicates the index of an interface.

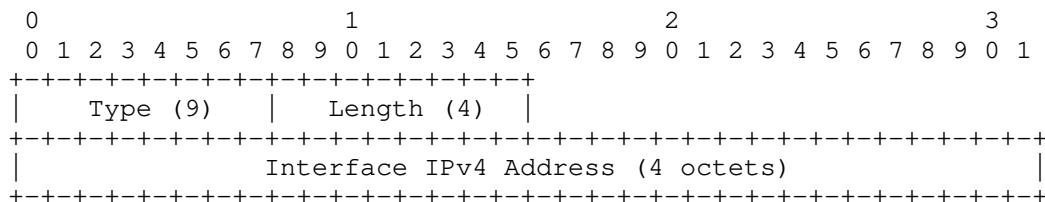


Figure 11: Interface IPv4 Address Sub-TLV

Type: Its value (9 suggested) is to be assigned by IANA.

Length: 4.

Interface IPv4 Address: 4 octets. It represents the IPv4 address of an interface.

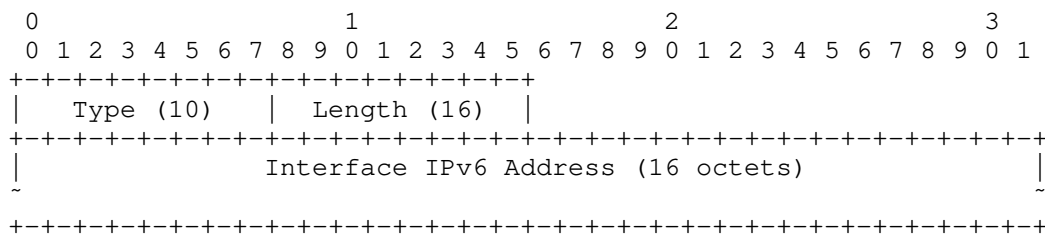


Figure 12: Interface IPv6 Address Sub-TLV

Type: Its value (10 suggested) is to be assigned by IANA.

Length: 16.

Interface IPv6 Address: 16 octets. It represents the IPv6 address of an interface.

6. Backup Ingress Behavior

When a backup ingress node receives a UPDATE message containing the information for protecting the primary ingress node of a SR path, it installs a forwarding entry in its FIB based on the information. The information is encoded in a SR policy of the following structure:

SR Policy SAFI NLRI: <Distinguisher, Policy-Color, Endpoint>

Attributes:

```

  Tunnel Encaps Attribute (23)
    Tunnel Type (15): SR Policy
      SR Path Ingress Protection Sub-TLV
        Primary Ingress Sub-TLV
        Service Sub-TLV
        Traffic Description Sub-TLV
      Preference Sub-TLV
      Binding SID Sub-TLV
      Explicit NULL Label Policy (ENLP) Sub-TLV
      Priority Sub-TLV
      Policy Name Sub-TLV
      Segment List Sub-TLV
        Weight Sub-TLV
        Segment Sub-TLV
        Segment Sub-TLV
      ...
    ...

```

Where:

- o SR Policy SAFI NLRI is defined in [I-D.ietf-idr-segment-routing-te-policy].
- o Tunnel Encapsulation Attribute is defined in [I-D.ietf-idr-tunnel-encaps].
- o Tunnel Type of SR Policy is defined in [I-D.ietf-idr-segment-routing-te-policy].
- o SR Path Ingress Protection, Primary Ingress, Service and Traffic Description Sub-TLVs are defined in this document.
- o Preference, Binding SID, ENLP, Priority, Policy Name, Segment List, Weight and Segment Sub-TLVs are defined in [I-D.ietf-idr-segment-routing-te-policy].

After receiving a SR policy with a SR Path Ingress Protection Sub-TLV, the backup ingress node will install one or more candidate paths into its "BGP table". Another module such as SRPM will choose one or more paths and install the forwarding entries for them in the data plane.

The forwarding entries for the paths installed in the data plane will be set to be inactive if the flag A in the SR Path Ingress Protection Sub-TLV is zero. When the primary ingress node fails, these forwarding entries are set to be active. The failure of the primary ingress may be detected by the backup ingress node through using a mechanism such as BFD. The IP address of the primary ingress in the Primary Ingress Sub-TLV may be used for detecting the failure of the primary ingress node.

If the flag A in the SR Path Ingress Protection Sub-TLV is one, then the forwarding entries for the paths installed in the data plane will be set to be active.

When there is a Service Sub-TLV in the SR Path Ingress Protection Sub-TLV, the ID or Label in the Service Sub-TLV will be included in the forwarding entries. When a packet is imported into a backup SR path using the forwarding entries, the service ID or Label is pushed first and then the sequence of segments represented in the Segment List Sub-TLV.

7. Security Considerations

Protocol extensions defined in this document do not affect the BGP security other than those as discussed in the Security Considerations section of [RFC5575].

8. Acknowledgements

The authors of this document would like to thank Dhruv Dhody for the comments.

9. IANA Considerations

9.1. BGP Tunnel Encapsulation Attribute Sub-TLVs

Under Existing Registry Name: "BGP Tunnel Encapsulation Attribute Sub-TLVs", IANA is requested to assign a new Sub-TLV value for SR Path Ingress Protection as follows:

Value	sub-TLV Name	Reference
-----	-----	-----
TBD1	SR Path Ingress Protection Sub-TLV	This Document

9.2. Ingress Protection Information Sub-TLVs

A new registry called "Ingress Protection Information Sub-TLVs" is defined in this document. IANA is requested to create and maintain new registry:

o Ingress Protection Information Sub-TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	sub-TLV Name	Reference
-----	-----	-----
0	Reserved	
1	Primary Ingress IPv4 Address Sub-TLV	This Document
2	Primary Ingress IPv6 Address Sub-TLV	This Document
3	Service Label Sub-TLV	This Document
4	32 Bits Service ID Sub-TLV	This Document
5	128 Bits Service ID Sub-TLV	This Document
6	IPv4 FEC Sub-TLV	This Document
7	IPv6 FEC Sub-TLV	This Document
8	Interface Index Sub-TLV	This Document
9	Interface IPv4 Address Sub-TLV	This Document
10	Interface IPv6 Address Sub-TLV	This Document
11-255	Unassigned	

10. References

10.1. Normative References

- [I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Talaulikar, K., Mattes, P., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", Work in Progress, Internet-Draft, draft-ietf-idr-segment-routing-te-policy-17, 14 April 2022, <<https://www.ietf.org/archive/id/draft-ietf-idr-segment-routing-te-policy-17.txt>>.
- [I-D.ietf-idr-tunnel-encaps]
Patel, K., Velde, G. V. D., Sangli, S. R., and J. Scudder, "The BGP Tunnel Encapsulation Attribute", Work in Progress, Internet-Draft, draft-ietf-idr-tunnel-encaps-22, 7 January 2021, <<https://www.ietf.org/archive/id/draft-ietf-idr-tunnel-encaps-22.txt>>.
- [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>>.
- [RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, DOI 10.17487/RFC7356, September 2014, <<https://www.rfc-editor.org/info/rfc7356>>.
- [RFC8424] Chen, H., Ed. and R. Torvi, Ed., "Extensions to RSVP-TE for Label Switched Path (LSP) Ingress Fast Reroute (FRR) Protection", RFC 8424, DOI 10.17487/RFC8424, August 2018, <<https://www.rfc-editor.org/info/rfc8424>>.

10.2. Informative References

- [I-D.bashandy-rtgwg-segment-routing-ti-lfa]
Bashandy, A., Filsfils, C., Decraene, B., Litkowski, S., Francois, P., Voyer, D., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", Work in Progress, Internet-Draft, draft-bashandy-rtgwg-segment-routing-ti-lfa-05, 4 October 2018, <<https://www.ietf.org/archive/id/draft-bashandy-rtgwg-segment-routing-ti-lfa-05.txt>>.
- [I-D.hegde-spring-node-protection-for-sr-te-paths]
Hegde, S., Bowers, C., Litkowski, S., Xu, X., and F. Xu, "Node Protection for SR-TE Paths", Work in Progress, Internet-Draft, draft-hegde-spring-node-protection-for-sr-te-paths-07, 30 July 2020, <<https://www.ietf.org/archive/id/draft-hegde-spring-node-protection-for-sr-te-paths-07.txt>>.

- [I-D.hu-spring-segment-routing-proxy-forwarding]
Hu, Z., Chen, H., Yao, J., Bowers, C., Yongqing, and
Yisong, "SR-TE Path Midpoint Restoration", Work in
Progress, Internet-Draft, draft-hu-spring-segment-routing-
proxy-forwarding-19, 11 April 2022,
<[https://www.ietf.org/archive/id/draft-hu-spring-segment-
routing-proxy-forwarding-19.txt](https://www.ietf.org/archive/id/draft-hu-spring-segment-routing-proxy-forwarding-19.txt)>.
- [I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and
P. Mattes, "Segment Routing Policy Architecture", Work in
Progress, Internet-Draft, draft-ietf-spring-segment-
routing-policy-22, 22 March 2022,
<[https://www.ietf.org/archive/id/draft-ietf-spring-
segment-routing-policy-22.txt](https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-22.txt)>.
- [I-D.sivabalan-pce-binding-label-sid]
Sivabalan, S., Filsfils, C., Tantsura, J., Hardwick, J.,
Previdi, S., and C. Li, "Carrying Binding Label/Segment-ID
in PCE-based Networks.", Work in Progress, Internet-Draft,
draft-sivabalan-pce-binding-label-sid-07, 8 July 2019,
<[https://www.ietf.org/archive/id/draft-sivabalan-pce-
binding-label-sid-07.txt](https://www.ietf.org/archive/id/draft-sivabalan-pce-binding-label-sid-07.txt)>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an
IANA Considerations Section in RFCs", RFC 5226,
DOI 10.17487/RFC5226, May 2008,
<<https://www.rfc-editor.org/info/rfc5226>>.
- [RFC5462] Andersson, L. and R. Asati, "Multiprotocol Label Switching
(MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic
Class" Field", RFC 5462, DOI 10.17487/RFC5462, February
2009, <<https://www.rfc-editor.org/info/rfc5462>>.
- [RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J.,
and D. McPherson, "Dissemination of Flow Specification
Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009,
<<https://www.rfc-editor.org/info/rfc5575>>.

Authors' Addresses

Huaimo Chen
Futurewei
Boston, MA,
United States of America
Email: huaimo.chen@futurewei.com

Mehmet Toy
Verizon
United States of America
Email: mehmet.toy@verizon.com

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing
102209
China
Email: wangaj3@chinatelecom.cn

Zhenqiang Li
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing
100053
China
Email: lizhengqiang@chinamobile.com

Lei Liu
Fujitsu
United States of America
Email: liulei.kddi@gmail.com

Xufeng Liu
Volta Networks
McLean, VA
United States of America
Email: xufeng.liu.ietf@gmail.com

Network Working Group
Internet Draft
Intended status: Informational
Expires: May 21, 2020
Consulting

L. Dunbar
Futurewei
S. Hares
Hickory Hill

November 21, 2019

SDWAN WAN Ports Property Advertisement in BGP UPDATE
draft-dunbar-idr-sdwan-port-safi-06

Abstract

The document describes how the SDWAN SAFI, which is assigned by IANA in the First Come First Server range, is used for SDWAN edge nodes to propagate its WAN port properties to its controller.

In the context of this document, BGP Route Reflectors (RR) is the component of the SDWAN Controller that receives the BGP UPDATE from SDWAN edges and in turns propagate the information to a group of authorized SDWAN edges reachable via overlay networks.

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>

This Internet-Draft will expire on Dec 5, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction.....	3
2. Conventions used in this document.....	3
2.1. Information to be propagated for SDWAN UPDATE.....	4
2.2. SAFI under the MP-NLRI.....	6
2.3. How about a new Path Attribute under BGP UPDATE?.....	6
3. SDWAN WAN Port Identifier encoding in the MP-NLRI Path Attribute	6
4. WAN Port Properties encoding in the Tunnel Path Attribute.....	8
4.1. Port Ext SubTLV for NAT.....	9
4.2. IPsec Security Association Property.....	10
4.3. Remote Endpoint.....	11
5. Manageability Considerations.....	12
6. Security Considerations.....	12
7. IANA Considerations.....	12
8. References.....	12
8.1. Normative References.....	12
8.2. Informative References.....	13
9. Acknowledgments.....	14

1. Introduction

[Net2Cloud-Problem] introduces using SDWAN to reach dynamic workloads in multiple third-party data centers and aggregate multiple underlay paths, including public untrusted networks, provided by different service providers.

[SDWAN-BGP-USAGE] describes multiple SDWAN scenarios and illustrates how BGP is used as control plane for the SDWAN networks.

The document describes BGP UPDATE for SDWAN edge nodes to propagate its WAN port properties to RR.

2. Conventions used in this document

Cloud DC: Off-Premise Data Centers that usually host applications and workload owned by different organizations or tenants.

Controller: Used interchangeably with SDWAN controller to manage SDWAN overlay path creation/deletion and monitor the path conditions between sites.

CPE-Based VPN: Virtual Private Secure network formed among CPEs. This is to differentiate from most commonly used PE-based VPNs a la RFC 4364.

MP-NLRI: The MP_REACH_NLRI Path Attribute defined in RFC4760.

SDWAN End-point: An WAN port (logical or physical) of a SDWAN edge node. (If "endpoint" is used, it refers to a SDWAN End-point).

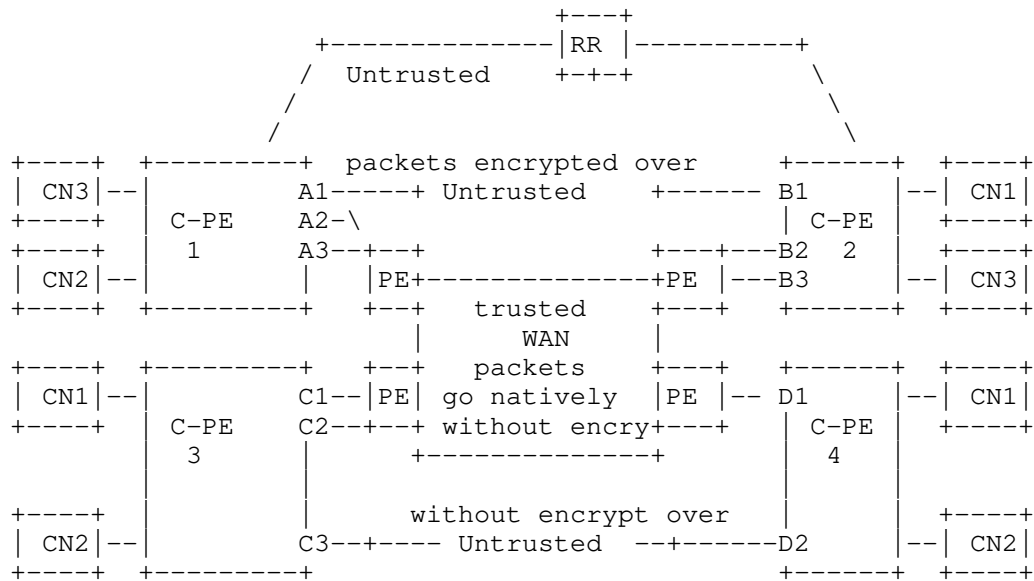
OnPrem: On Premises data centers and branch offices

SDWAN: Software Defined Wide Area Network. In this document, "SDWAN" refers to the solutions of pooling WAN bandwidth from multiple underlay networks to get better WAN bandwidth management, visibility & control. When the underlay networks are private networks, traffic can be

forwarded without additional encryption; when the underlay networks are public, such as Internet, some traffic needs to be encrypted when forwarding through those WAN ports (depending on user provided policies).

2.1. Information to be propagated for SDWAN UPDATE

Figure below shows the Hybrid SDWAN scenario:



CN: Client Network

Figure 1: Hybrid SDWAN

Using C-PE2 for illustration, C-PE2 needs to send out two separate BGP UPDATE messages.

BGP UPDATE #1 is to propagate C-PE2 attached routes, which are the regular VPN (L3VPN or EVPN) BGP Route UPDATE message,

MP-NLRI Path Attribute

```
Nexthop (C-PE2)
NLRI
  10.1.x.x.
  VLAN 15
  12.1.1x
Tunnel-Encap Path Attribute
  Details of any tunnels that applicable to the routes carried
  by the MP-NLRI Path Attribute
```

BGP UPDATE #2 is to propagate C-PE2's WAN port properties to RR, which should include:

- Identifier for the WAN Port
- The NAT property for the WAN Port
- The minimum IPsec information for establishing Port based IPsec.

Separating WAN port properties UPDATE from client routes UPDATE makes the implementation simpler, because the properties of a SDWAN node's WAN Port can change independent from the client routes attached to the C-PE2. WAN port properties change can be caused by many factors, such as ISP service agreement changes for the service connected to the WAN Port, the WAN port being disabled, or its IPsec property changes, etc. Since most SDWAN edges only have a small number of WAN ports, the disadvantage of multiple BGP UPDATE messages to advertise properties of those WAN ports is relatively small.

Following the same approach used by [idr-segment-routing-te-policy] where the SR Policy identifier is encoded in the MP-NLRI Path Attribute and the detailed SR Policies are encoded in the Tunnel Path attribute, the BGP UPDATE for SDWAN WAN port can have the WAN Port identifier encoded in the MP-NLRI Path Attribute and the associated WAN Port properties encoded in the Tunnel Path Attribute.

Receivers of the UPDATE can associate the SDWAN node identifier, site identifier with the node's WAN Port properties.

2.2. SAFI under the MP-NLRI

It is possible to continue using the same IP SAFI in the MP-NLRI [RFC4760] Path Attribute for advertising the SDWAN WAN port properties. If the same IP SAFI used, receiver needs extra logic to differentiate regular BGP MP-NLRI routes advertisement from the SDWAN WAN port properties advertisement and recognize the extra Site ID field added to the MP-NLRI. The benefit of using the same IP SAFI is that the UPDATE can traverse existing routers without being dropped. However, the SDWAN UPDATE is only between SDWAN edge and the RR, all the intermediate nodes treat the UPDATE message as regular IP data frame.

That is why it is simpler to follow the same approach used by [idr-segment-routing-te-policy] to have a unique SAFI (IANA assigned SDWAN SAFI = 74) mainly to differentiate the SDWAN UPDATE from regular route UPDATE.

This SDWAN SAFI is for a scenario where one SDWAN edge node has multiple WAN ports, some of which connected to private networks and others connected to public untrusted networks [Scenario #2 described in the [SDWAN-BGP-USAGE]]. The same routes attached to the SDWAN can be reached by the private networks without encryption (for better performance) or by the public networks with encryption.

2.3. How about a new Path Attribute under BGP UPDATE?

It is also possible to have a new Path Attribute, say SDWAN Path Attribute, combined with Tunnel Path Attribute to advertise SDWAN WAN Port properties. Besides having a different Path Attribute ID, everything else is same as using MP-NLRI & Tunnel Path Attributes.

3. SDWAN WAN Port Identifier encoding in the MP-NLRI Path Attribute

SDWAN WAN Port Identifier needs the following attributes

- locally significant port number,
- the location of the SDWAN device, and
- the globally routable address for the WAN Port.

Here is the encoding for those attributes in the NLRI field within the MP_REACH_NLRI Path Attribute of RFC4760, under a SDWAN SAFI (code = 74):

-----+		
	NLRI Length	1 octet
-----+		
	SDWAN-Type	2 Octets
-----+		
	Port-Local-ID	4 octets
-----+		
	SDWAN-Site-ID	4 octets
-----+		
	SDWAN-Node-ID	4 or 16 octets
-----+		

where:

- NLRI Length: 1 octet of length expressed in bits as defined in [RFC4760].
- SDWAN-Type: to define the encoding of the rest of the SDWAN NLRI. There could be different sub-TLVs for different SDWAN WAN ports and their associated policies.
- Port local ID: SDWAN edge node Port identifier, which can be locally significant. Each port can have unique properties. For example, some ports may get ISP or DHCP assigned IP addresses (IPv4 or IPv6), some may have private IP addresses that packets to/from those ports have to traverse NAT. The detailed properties about the port are further encoded in the subTLVs, e.g. Port-subTLV under the Tunnel Path Attribute.
- SDWAN-Site-ID: used to identify a common property shared by a set of SDWAN edge nodes, such as the property of a specific geographic location shared by a group of SDWAN edge nodes. The property is used to steer an overlay route to traverse specific geographic locations for various reasons, such as to comply

regulatory rules, to utilize specific value added services, or others.

- SDWAN EdgeNode ID: the SDWAN edge node identifier, which has to be a routable address (IPv4 or IPv6) within the WAN.

4. WAN Port Properties encoding in the Tunnel Path Attribute

The content of the SDWAN Port properties is encoded in the Tunnel Encapsulation Attribute defined in [Tunnel-Encap] using a new Tunnel-Type TLV (code point to be assigned by IANA from the "BGP Tunnel Encapsulation Attribute Tunnel Types" registry).

Tunnel Encaps Path Attribute (Code = 23)

Tunnel Type: SDWAN-WAN-Port

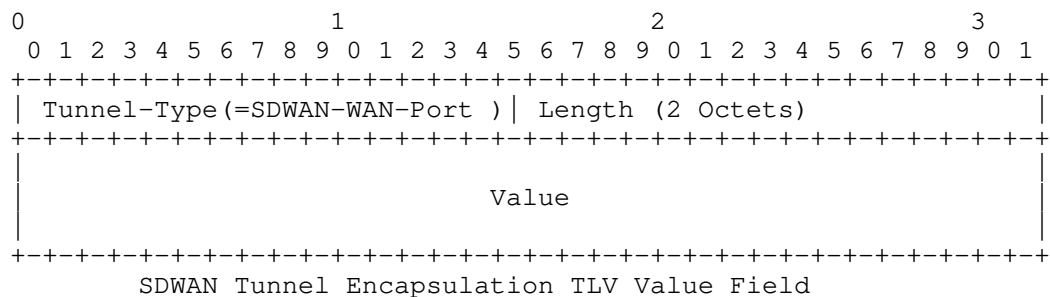
Followed by the detailed properties encoded as subTLV, such as

SubTLV for NAT

SubTLV for IPsec-SA Attribute

SubTLV for ISP connected to the WAN port

The Tunnel Encaps Attribute are defined as follows:



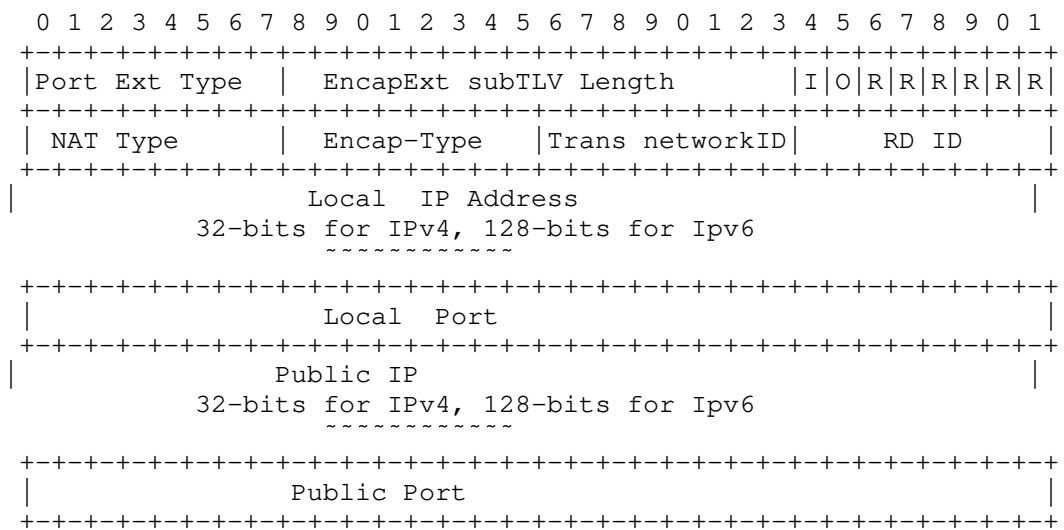
Where:

Tunnel Type is SDWAN-WAN-Port (to be assigned by IANA).

4.1. Port Ext SubTLV for NAT

NAT information is encoded is the Port Ext sub-TLV is for describing the NAT property if the port has private address and the network identifier to which the WAN port is connected, etc.

A SDWAN edge node can inquire STUN (Session Traversal of UDP Through Network Address Translation RFC 3489) Server to get the NAT property, the public IP address and the Public Port number to pass to peers.



Where:

- o Port Ext Type: indicate it is the Port Ext SubTLV.
- o PortExt subTLV Length: the length of the subTLV.
- o Flags:
 - I bit (CPE port address or Inner address scheme)
 - If set to 0, indicate the inner (private) address is IPv4.
 - If set to 1, it indicates the inner address is IPv6.
 - O bit (Outer address scheme):
 - If set to 0, indicate the public (outer) address is IPv4.

If set to 1, it indicates the public (outer) address is IPv6.

- R bits: reserved for future use. Must be set to 0 now.

- o NAT Type.without NAT; 1:1 static NAT; Full Cone; Restricted Cone; Port Restricted Cone; Symmetric; or Unknown (i.e. no response from the STUN server).
- o Encap Type.the supported encapsulation types for the port facing public network, such as IPsec+GRE, IPsec+VxLAN, IPsec without GRE, GRE (when packets don't need encryption)
- o Transport Network ID.Central Controller assign a global unique ID to each transport network.
- o RD ID.Routing Domain ID.Need to be global unique.
- o Local IP.The local (or private) IP address of the port.
- o Local Port.used by Remote SDWAN edge node for establishing IPsec to this specific port.
- o Public IP.The IP address after the NAT. If NAT is not used, this field is set to NULL.
- o Public Port.The Port after the NAT. If NAT is not used, this field is set to NULL.

4.2. IPsec Security Association Property

The IPsecSA sub-TLV is for the SDWAN edge node to establish IPsec security association with their peers via the port that face untrusted network. The minimum set of the IPsec information is from [CONTROLLER-IKE].

```

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|IPsec-SA Type |IPsecSA Length|Flag|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Transform    | Transport    | AH      | ESP      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Key Counter  |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| key1 length  | Public Key |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| key2 length  | Nonce      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| key3 length | key3 (for potential other keys |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|           Duration           |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

```

Where:

- o IPsec-SA SubTLV Type: to be assigned by IANA. The type value has to be between 128~255 because IPsec-SA subTLV needs 2 bytes for length to carry the needed information.
- o IPsec-SA subTLV Length (2 Byte): 25 (or more)
- o Flags: 1 octet of flags. None are defined at this stage. Flags SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Transform (1 Byte): the value can be AH, ESP, or AH+ESP.
- o Transport (1 byte): the value can be Tunnel Mode or Transport mode
- o AH (1 byte): AH authentication algorithms supported, which can be md5 | sha1 | sha2-256 | sha2-384 | sha2-512 | sm3. Each SDWAN edge node can have multiple authentication algorithms; send to its peers to negotiate the strongest one.
- o ESP (1 byte): ESP authentication algorithms supported, which can be md5 | sha1 | sha2-256 | sha2-384 | sha2-512 | sm3. Each SDWAN edge node can have multiple authentication algorithms; send to its peers to negotiate the strongest one. Default algorithm is AES-256.
- o Rekey Counter: 4 bytes
- o Public Key: IPsec public key
- o Nonce: IPsec Nonce
- o Key3: other potential key
- o Duration: SA life span.

4.3. Remote Endpoint

The Remote Endpoint sub-TLV is not used for SDWAN NLRI because

- o The SDWAN Node ID and Site ID are already encoded in the SDWAN NLRI,
- o The network connected by the SDWAN WAN port might have identifier that is more than the AS number. SDWAN controller might use its own specific identifier for the network.

- o The Transport-Network-ID in the EncapExt sub-TLV represents the SDWAN unique network identifier.

If the Remote Endpoint Sub-TLV is present, it is ignored by other SDWAN edge nodes.

5. Manageability Considerations

TBD - this needs to be filled out before publishing

6. Security Considerations

The document describes the encoding for SDWAN edge nodes to advertise its SDWAN WAN ports properties to their peers via untrusted & unsecure networks.

The secure propagation is achieved by secure channels, such as TLS, SSL, or IPsec, between the SDWAN edge nodes and the local controller RR.

[More details need to be filled in here]

7. IANA Considerations

This document requires the following IANA actions.

- o SDWAN Overlay SAFI = 74 assigned by IANA
- o SDWAN Route Type

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

8.2. Informative References

- [RFC8192] S. Hares, et al, "Interface to Network Security Functions (I2NSF) Problem Statement and Use Cases", July 2017
- [RFC5521] P. Mohapatra, E. Rosen, "The BGP Encapsulation Subsequent Address Family Identifier (SAFI) and the BGP Tunnel Encapsulation Attribute", April 2009.
- [CONTROLLER-IKE] D. Carrel, et al, "IPsec Key Exchange using a Controller", draft-carrel-ipsecme-controller-ike-01, work-in-progress.
- [Tunnel-Encap] E. Rosen, et al, "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-09, Feb 2018.
- [VPN-over-Internet] E. Rosen, "Provide Secure Layer L3VPNs over Public Infrastructure", draft-rosen-bess-secure-l3vpn-00, work-in-progress, July 2018
- [DMVPN] Dynamic Multi-point VPN:
<https://www.cisco.com/c/en/us/products/security/dynamic-multipoint-vpn-dmvpn/index.html>
- [DSVPN] Dynamic Smart VPN:
<http://forum.huawei.com/enterprise/en/thread-390771-1-1.html>
- [ITU-T-X1036] ITU-T Recommendation X.1036, "Framework for creation, storage, distribution and enforcement of policies for network security", Nov 2007.
- [Net2Cloud-Problem] L. Dunbar and A. Malis, "Seamless Interconnect Underlay to Cloud Overlay Problem Statement", draft-dm-net2cloud-problem-statement-02, June 2018
- [Net2Cloud-gap] L. Dunbar, A. Malis, and C. Jacquenet, "Gap Analysis of Interconnecting Underlay with Cloud Overlay", draft-dm-net2cloud-gap-analysis-02, work-in-progress, Aug 2018.

[Tunnel-Encap] E. Rosen, et al "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-10, Aug 2018.

9. Acknowledgments

Acknowledgements to Wang Haibo, Hao Weiguo, and ShengCheng for implementation contribution; Many thanks to Jim Guichard, John Scudder, Darren Dukes, Andy Malis, Rachel Huang and Donald Eastlake for their review and contributions.

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Linda Dunbar
Futurewei
Email: ldunbar@futurewei.com

Sue Hares
Hickory Hill Consulting
Email: shares@ndzh.com

idr
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2020

J. Hu
Nokia
March 9, 2020

BGP Provisioned IPsec Tunnel Configuration
draft-hujun-idr-bgp-ipsec-02

Abstract

This document defines a method of using BGP to provide IPsec tunnel configuration along with NLRI, it uses and extends tunnel encapsulation attribute as specified in [I-D.ietf-idr-tunnel-encaps] for IPsec tunnel.

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 September 10, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction	2
1.1. Terminology	3
2. Tunnel Encapsulation Attribute for IPsec	3
2.1. Local and Remote Prefix sub-TLV	4
2.2. Public Routing Instance sub-TLV	5
2.3. IPsec Configuration Tag sub-TLV	5
3. Operation	6
4. Semantics and Usage of IPsec Tunnel Encapsulation attribute .	10
4.1. Nested Tunnel	10
4.2. Other Operation Specifics	11
5. IANA Considerations	11
6. Security Considerations	12
7. Change Log	13
8. References	13
8.1. Normative References	13
8.2. Informative References	14
Author's Address	15

1. Introduction

IPsec is the standard for IP layer traffic protection, however in a big network where mesh connections are needed, configuring large number of IPsec tunnels is error prone and not scalable. So instead of pre-provision IPsec tunnels on each router, this document defines a method to allow router to advertise the IPsec tunnel configurations it requires to reach a given NLRI via BGP. This document does not intend to be one solution for all cases, the main use case is to simplify IPsec tunnel provision in networks under single administrative domain; it uses standard based components (IPsec/IKEv2[RFC7296] and BGP) with limited changes. There is no change to IPsec/IKEv2, and only limited changes to BGP.

IPsec tunnel in this document means IPsec tunnel mode as defined in [RFC4301].

IPsec tunnel configurations typically include following parts:

- o tunnel endpoint address (local and remote)
- o public routing instance, routing instance where IPsec packet is forwarded in
- o private routing instance, routing instance where payload packet is forwarded in
- o tunnel authentication method and credentials

- o IKE SA and CHILD SA transform (a.k.a crypto algorithms)
- o CHILD SA traffic selector
- o other: like lifetime, DPD timer, use of PFS ..etc

In order to minimize amount configurations signal via BGP, only following configurations are explicit advertised:

- o local tunnel endpoint address: BGP tunnel encapsulation attribute
- o public routing instance: sub-TLV in tunnel encapsulation attribute
- o CHILD SA traffic selector address range: NLRI and/or sub-TLV in tunnel encapsulation attribute

Other configurations are either derived or via tag mapping:

- o remote tunnel endpoint address: dynamic learned when received IKEv2 IKE_SA_INIT request
- o private routing instance: via route-target in same BGP UPDATE
- o tunnel authentication/credentials, traffic selector protocol/port range, IKE SA and CHILD SA transform, lifetime, DPD timer, PFS ..etc: all these configurations are implicitly signaled via IPsec configuration tag sub-TLV in tunnel encapsulation attribute

[I-D.ietf-idr-tunnel-encaps] defines a generic tunnel encapsulation attribute for BGP, however it needs to be extended to support IPsec tunnel.

1.1. 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.

2. Tunnel Encapsulation Attribute for IPsec

This document extends tunnel encapsulation attribute specified in [I-D.ietf-idr-tunnel-encaps] by introducing following changes:

- o A tunnel type for IPsec tunnel: ESP tunnel mode (AH tunnel mode is not included in this document). Existing type 4 (IPsec in Tunnel-

mode) in IANA "BGP Tunnel Encapsulation Attribute Tunnel Types" registry could be reused

- o A new sub-TLV for public routing instance
- o A new sub-TLV for remote address prefix
- o A new sub-TLV for local address prefix
- o A new sub-TLV for IPsec configuration tag

Following existing sub-TLVs apply to IPsec tunnel encapsulation attribute:

- o Remote Endpoint: IPsec tunnel endpoint address
- o Embedded Label Handling: see Section 4 for detail

2.1. Local and Remote Prefix sub-TLV

Local prefix sub-TLV is an optional sub-TLV used to specify a list of address prefix that used as local traffic selector address ranges; if local prefix sub-TLV is not included, then prefixes in NLRI will be used; Remote prefix sub-TLV is a mandatory sub-TLV used to specify a list of address prefix that used as remote traffic selector address ranges; The IP version of local/remote prefix MUST be as same as IP version of prefix in NLRI. A single all zero prefix means any prefix is allowed. Local and remote prefix sub-TLV has same encoding as following:

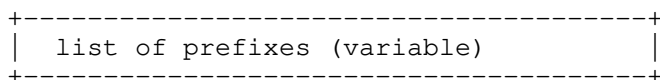


Figure 1: Source Prefix sub-TLV

Each prefix is encoded as following:

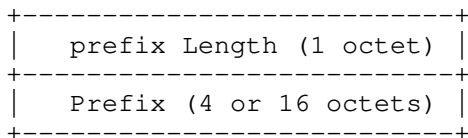


Figure 2: prefix

For a given IPsec tunnel TLV, local prefix sub-TLV MUST appear either zero or one time; remote prefix sub-TLV MUST appear only one time.

2.2. Public Routing Instance sub-TLV

Public routing instance sub-TLV is an optional sub-TLV used to specify the routing instance to which the remote point address belongs, if tunnel encapsulation attribute doesn't include this TLV, then the routing instance is the same to which BGP session belongs. the value field of the sub-TLV consist a route target community as defined in [RFC4360].

For a given IPsec tunnel TLV, public routing instance sub-TLV MUST appear either zero or one time.

2.3. IPsec Configuration Tag sub-TLV

This sub-TLV represents the IPsec configurations (like IPsec transform) that are not explicit advertised by other sub-TLVs specified in this documentation; the meaning of this sub-TLV is local to the administrative domain. Follow are some examples:

- o tag value T1 map to following configurations:
 - * Certificate trust-anchor: CA-1
 - * IKE_SA/CHILD_SA transform: AES-GCM-128
 - * Diffie-Hellman Group: 15
 - * Perfect Forward Secrecy: No
 - * local/remote Traffic selector protocol: any
 - * local/remote Traffic selector port range: any
 - * IKE_SA lifetime: 24 hours
 - * CHILD_SA lifetime: 1 hour
 - * DPD interval: 30 seconds
 - * ESP extended sequence number: no
- o tag value T2 map to following configurations:
 - * Certificate trust-anchor: CA-2
 - * IKE_SA/CHILD_SA transform: AES-GCM-256
 - * Diffie-Hellman Group: 20

- * Perfect Forward Secrecy: Yes with group 20
- * local/remote Traffic selector protocol: UDP
- * local/remote Traffic selector port range: any
- * IKE_SA lifetime: 48 hours
- * CHILD_SA lifetime: 2 hours
- * DPD interval: 10 seconds
- * ESP extended sequence number: yes

The value field of this sub-TLV is 4 octets long. each IPsec tunnel TLV SHOULD only contain one IPsec configuration tag sub-TLV;

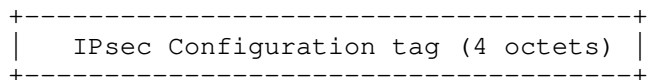


Figure 3: IPsec Configuration Tag

For a given IPsec tunnel TLV, IPsec configuration tag sub-TLV MUST appear only one time.

3. Operation

Following are the rules of operation:

1. All routers are in same administrative domain
2. All routers are pre-provisioned with Mapping between IPsec configuration tag value and IPsec configurations include authentication method/credentials
3. If a given NLRI need IPsec protection, then advertising router need to include an IPsec tunnel encapsulation attribute, along with the NLRI in BGP UPDATE U;
4. When a router need to forward a packet along a path is determined by a BGP UPDATE which has a tunnel encapsulation attribute that contains one or more IPsec tunnel TLV, and router decides use IPsec based on local policy, then the router use first feasible CHILD_SA, a CHILD SA is considered as feasible when it meets all following conditions:

- * its private routing instance is same as routing instance to which the packet to be forwarded belongs
 - * its public routing instance is same as indicated by the Public Routing Instance sub-TLV; if the sub-TLV doesn't exist, then it is same as routing instance to which BGP session belongs
 - * its peer tunnel address is same as indicated by Remote Endpoint sub-TLV
 - * the source and destination address of the packet to be forwarded falls in the range of CHILD SA's traffic selector
 - * its transform and other configuration maps to the tag indicated in the IPsec configuration tag sub-TLV
5. If router can't find such CHILD SA, then it will use IKEv2 to create one; if there are multiple IPsec tunnel TLVs in U, then it need to select one from feasible TLVs, a IPsec tunnel TLV is considered as feasible when it meets all following requirements:
- * the source address of the packet must fall in one of Remote Prefixes
 - * the destination address of the packet must fall one of Source Prefixes
 - * the Remote Endpoint, along with Public Routing Instance sub-TLV identifies an IP address that is reachable
6. If there are multiple feasible IPsec tunnel TLV exists, then select the TLV using following rules in order:
1. TLV with smallest local address range as indicated by Remote Prefix sub-TLV
 2. TLV with smallest remote address range as indicated by Local Prefix sub-TLV (NLRI prefix if local prefix sub-TLV is not included in TLV)
7. After an IPsec TLV is selected, router uses IKEv2 to create the CHILD_SA:
- * public/private routing instance, peer's tunnel address are chosen based on above rules
 - * Traffic Selector:

- * For each TS in TSi:
 - + address range: the prefix specified in Remote Prefix sub-TLV
 - + protocol: tag mapped configuration
 - + port range: tag mapped configuration
- * for each TS in TSr:
 - + address range: prefixes specified by Local Prefix sub-TLV if it exists; otherwise use the prefix specified by the NLRI
 - + protocol: tag mapped configuration
 - + port range: tag mapped configuration

The operation of BGP provisioned IPsec configuration is illustrated with following example:

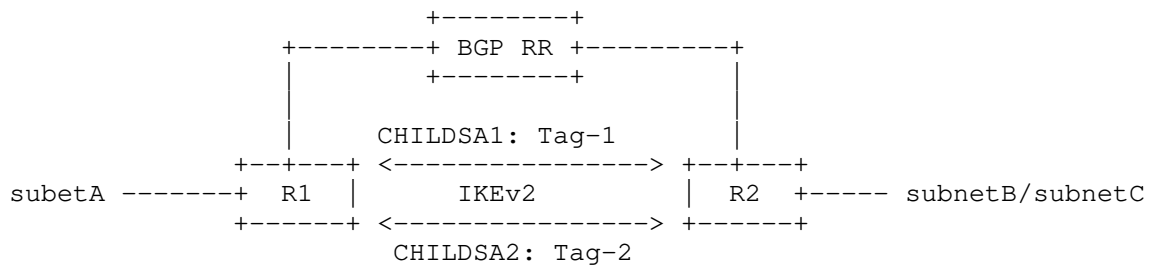


Figure 4: Operation Example

There are following traffic protection requirements:

- o subnetA - subnetB: ESP tunnel, CHACHA20_POLY1305 , mapping to tag Tag-1
- o subnetA - subnetC: ESP tunnel, NULL-AES-GMAC-256 , mapping to tag Tag-2
- o note: other IPsec configurations, like IKE_SA lifetime ..etc, are the same for both Tag-1 and Tag-2; not listed here for sake of

Both R1 and R2 are provisioned with IPsec authentication credentials and configurations corresponding to Tag-1 and Tag-2; both Tag-1 and Tag-2 map to traffic selector protocol any and port range any.

- o R1 advertise subnetA in BGP UPDATE, which has a tunnel encapsulation attribute that contains two IPsec tunnel TLVs:
 - * TLV-1: endpoint R1TunnelAddr, tag sub-TLV Tag-1 and subnetB in Remote Prefix sub-TLV.
 - * TLV-2: endpoint R1TunnelAddr, tag sub-TLV Tag-2 and subnetC in Remote Prefix sub-TLV.
- o R2 advertise subnetB in BGP UPDATE, which has a tunnel encapsulation attribute that contains one IPsec tunnel TLV: R2TunnelAddr, tag sub-TLV Tag-1 and subnetA in Remote Prefix sub-TLV.
- o R2 advertise subnetC in BGP UPDATE, which has a tunnel encapsulation attribute that contains one IPsec tunnel TLV: R2TunnelAddr, tag sub-TLV Tag-2 and subnetA in Remote Prefix sub-TLV.
- o R1 received a packet from subnetA destined to subnetB, since BGP UPDATE contain subnetB also contains an IPsec tunnel encapsulation attribute, there is no existing CHILD SA could be used, based on the rules described in this section, R1 select TLV-1 and uses IKEv2 to establish an IPsec tunnel to R2TunnelAddr, using certificate authentication, create 1st CHILD SA CHILDSA1:
 - * ESP transform: CHACHA20_POLY1305
 - * Traffic Selector:
 - + TSi: address subnetA, protocol any, port any
 - + TSr: address subnetB, protocol any, port any
- o after tunnel is created, R1 and R2 could forward traffic between subnetA and subnetB over CHILDSA1
- o R1 received a packet from subnetA destined to subnetC, CHILDSA1 can't be used for this packet, R1 select TLV-2 to create 2nd CHILD SA, and given there is already an IKE SA between R1 and R2, R1 uses existing IKESA to create CHILDSA2:
 - * ESP transform: NULL-AES-GMAC-256

* Traffic Selector:

- + TSi: address subnetA, protocol any, port any
- + TSr: address subnetC, protocol any, port any
- o R1 and R2 could forward traffic between subnetA and subnetC over CHILDSA2

4. Semantics and Usage of IPsec Tunnel Encapsulation attribute

IPsec tunnel encapsulation TLV has same usage and semantics as defined in [I-D.ietf-idr-tunnel-encaps] with following specific to IPsec tunnel:

- o Due to nature of IPsec, the payload packet could only be IPv4 or IPv6 packet, so it MAY be carried in any BGP UPDATE message whose AFI/SAFI is 1/1 (IPv4 Unicast), 2/1 (IPv6 Unicast).
- o For 1/128 (VPN-IPv4 Labeled Unicast), 2/128 (VPN-IPv6 Labeled Unicast), these NLRI has embedded label, which cause the payload packet can't be encapsulated in ESP packet, however with IPsec tunnel encapsulation, the label could be ignored during encapsulation since CHILD SA itself could be used to identify the private routing instance; so an UPDATE that include IPsec tunnel encapsulation attribute, which contains value 2 of Embedded Label Handling Sub-TLV, could be used to signal this type of setup.
- o For other types of AFI/SAFI, a nested tunnel setup could be used to get IPsec protection, for example, an 25/70 (EVPN) payload packet could be encapsulated in VXLAN over IPsec tunnel. See Section 4.1 for further detail.

4.1. Nested Tunnel

A nested tunnel could be used for payload packet type that can't be encapsulated in IPsec tunnel directly, e.g. an Ethernet packet of EVPN service. Following is an example of using VXLAN over IPsec tunnel for EVPN service:

- o R1 need to forward an Ethernet packet P
- o the path along which P is to be forwarded is determined by BGP UPDATE U1, which has a VXLAN tunnel encapsulation attribute and the next-hop is router R2
- o the best path to R2 is a BGP route that was advertised in BGP UPDATE U2, which has an IPsec tunnel encapsulation TLV.

- o R1 will encapsulate P in a VXLAN tunnel as indicated in U1, then encapsulate VXLAN packet into IPsec tunnel as indicated in U2
- o if tag sub-TLV is used, then both U1 and U2 MUST have matching tag sub-TLV, otherwise the VXLAN packet will not be sent through IPsec tunnels identified in U2

4.2. Other Operation Specifics

Following are some operation specific rules:

1. An IPsec dead peer detection mechanism, like IKEv2 DPD or BFD over IPsec, SHOULD be used to monitor liveness of IPsec tunnel;
2. If IPsec peer goes down, as described in section 5 of [I-D.ietf-idr-tunnel-encaps], packet forwarding router chooses another functional tunnel, specified by another tunnel TLV of same BGP route if there is any, to forward the packet; if there is no such tunnel, then router MAY drop the packet or MAY forward packet as it would had the Tunnel Encapsulation attribute not been present. this is matter of local policy.
3. After IPsec peer goes down, packet forwarding router SHOULD try to re-establish IPsec tunnel with certain hold-down timer and back-off mechanism. the detail is up to implementation. also IKEv2 session resumption [RFC5723] MAY be used to efficiently re-create tunnel;
4. When router receives a packet destined to a BGP route it advertised but does not have any of tunnel encapsulation in the BGP route, it MAY drop it or MAY accept it; this is matter of local policy. by default, the packet should be accepted.
5. As with all types of tunnel technology, IPsec tunnel adds overhead (crypto & encapsulation) to the packet, which often causes MTU issues, deployment SHOULD take tunnel overhead into MTU consideration.

5. IANA Considerations

This document reuses "IPsec in Tunnel-mode"(4) as BGP Tunnel Encapsulation Attribute Tunnel Types.

This document will request new values in IANA "BGP Tunnel Encapsulation Attribute Sub-TLVs" registry for following sub-TLV:

- o public routing instance
- o remote address prefix
- o local address prefix
- o IPsec configuration tag

6. Security Considerations

IKEv2 is used to create IPsec tunnel, which ensures following:

- o Traffic protection keys are generated dynamically during IKEv2 negotiation, only known by participating peer of the IPsec tunnel; there is no central node to manage and distribute all keys.
- o IKEv2 rekey mechanism refresh keys regularly; PFS(Perfect Forward Secrecy) provides additional protection;
- o Secure authentication mechanism that only allow authenticated peer to create tunnel
- o Traffic Selector guarantee that only agreed traffic is allowed to be forwarded within the IPsec tunnel;
- o Using a separate, dedicate protocol(IKEv2) for key management/ authentication ensure they are not tied to BGP, all existing and future IKEv2 features could be used without changing BGP;

There is concern that malicious party might manipulate IPsec tunnel encapsulation attribute to divert traffic, however this risk could be mitigated by IKEv2 mutual authentication.

BGP route filter include outbound route filter [RFC5291], Origin Validation [RFC6811] and BGPSec [RFC8205] could be used to further secure BGP UPDATE message.

IKEv2 cookie [RFC7296] and varies mechanisms defined including client puzzle defined in [RFC8019] could be used to protect IKEv2 from Distributed Denial-of-Service Attacks.

Follow latest IETF ESP/IKEv2 implementation requirement and guidance ([RFC8221] and [RFC8247] at time of writing) to make sure always using secure and up-to-date cryptographic algorithms;

7. Change Log

- o v00 March 04, 2019: initial draft
- o v01 Sep 04, 2019:
 - * replaces color sub-TLV with a new IPsec configuration tag sub-TLV
 - * add rule on selecting TLV when there multiple feasible TLVs in section Section 3
 - * change crypto used in example of section Section 3
 - * change title from "BGP Signaled IPsec Tunnel Configuration" to "BGP Provisioned IPsec Tunnel Configuration"
 - * Add a section Section 4.2 on some operation specifics
 - * add more content in Section 6
 - * add specification of number of time each new sub-TLV allowed in a given tunnel TLV
 - * add clarification in section Section 1 to clarify IPsec tunnel means IPsec tunnel mode
 - * traffic selector protocol and port range now come from tag mapped configuration
- o v02 March 09, 2020
 - * increase version number to keep draft afloat

8. References

8.1. Normative References

- [I-D.ietf-idr-tunnel-encaps]
Patel, K., Velde, G., and S. Ramachandra, "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-15 (work in progress), December 2019.
- [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>>.

- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <<https://www.rfc-editor.org/info/rfc4301>>.
- [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>>.
- [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>>.

8.2. Informative References

- [RFC5291] Chen, E. and Y. Rekhter, "Outbound Route Filtering Capability for BGP-4", RFC 5291, DOI 10.17487/RFC5291, August 2008, <<https://www.rfc-editor.org/info/rfc5291>>.
- [RFC5723] Sheffer, Y. and H. Tschofenig, "Internet Key Exchange Protocol Version 2 (IKEv2) Session Resumption", RFC 5723, DOI 10.17487/RFC5723, January 2010, <<https://www.rfc-editor.org/info/rfc5723>>.
- [RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", RFC 6811, DOI 10.17487/RFC6811, January 2013, <<https://www.rfc-editor.org/info/rfc6811>>.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <<https://www.rfc-editor.org/info/rfc7296>>.
- [RFC8019] Nir, Y. and V. Smyslov, "Protecting Internet Key Exchange Protocol Version 2 (IKEv2) Implementations from Distributed Denial-of-Service Attacks", RFC 8019, DOI 10.17487/RFC8019, November 2016, <<https://www.rfc-editor.org/info/rfc8019>>.
- [RFC8205] Lepinski, M., Ed. and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <<https://www.rfc-editor.org/info/rfc8205>>.

- [RFC8221] Wouters, P., Migault, D., Mattsson, J., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 8221, DOI 10.17487/RFC8221, October 2017, <<https://www.rfc-editor.org/info/rfc8221>>.
- [RFC8247] Nir, Y., Kivinen, T., Wouters, P., and D. Migault, "Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 8247, DOI 10.17487/RFC8247, September 2017, <<https://www.rfc-editor.org/info/rfc8247>>.

Author's Address

Hu Jun
Nokia
777 East Middlefield Road
Mountain View CA 95148
United States

Email: jun.hu@nokia.com

idr
Internet-Draft
Intended status: Standards Track
Expires: April 12, 2020

J. Hu
Nokia
October 10, 2019

BGP Provisioned IPsec Transport Mode Protected Tunnel Configuration
draft-hujun-idr-bgp-ipsec-transport-mode-00

Abstract

This document defines a method of using BGP to advertise IPsec transport mode protected tunnel (like GRE tunnel with IPsec transport mode protection) configuration along with NLRI, based on [I-D.ietf-idr-tunnel-encaps] and [I-D.hujun-idr-bgp-ipsec].

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 April 12, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction	2
1.1. Terminology	2
1.2. IPsec Transport Protected sub-TLV	3
2. Semantics and Operation	3
3. IANA Considerations	5
4. Security Considerations	5
5. Change Log	6
6. References	6
6.1. Normative References	6
6.2. Informative References	6
Author's Address	7

1. Introduction

[I-D.hujun-idr-bgp-ipsec] defines a method of using BGP to advertise configuration for IPsec tunnel with ESP tunnel mode, however there are other use cases require of using IPsec/ESP transport mode with other types of IP tunnel, like GRE tunnel, as defined in [RFC4301] and [RFC4303]. Figure 2 shows an example of IPv4 GRE tunnel packet with ESP transport mode protection. This document defines a method of using BGP to advertise configuration for these use cases.

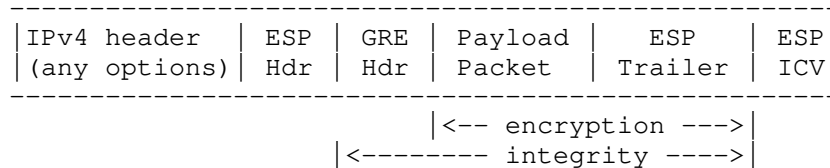


Figure 1: IPv4 GRE tunnel packet with ESP transport protection

The method follows same principle as [I-D.hujun-idr-bgp-ipsec], keep changes to BGP minimal and not changing IKEv2/IPsec; however the IPsec transport mode protected IP tunnel is not a tunnel stack or nested tunnels, IPsec transport mode protection doesn't add extra IP header.

The requirement of using IPsec transport mode is signaled by including a sub-TLV: IPsec transport protected, in a BGP tunnel encapsulation TLV.

1.1. Terminology

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

14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. IPsec Transport Protected sub-TLV

This sub-TLV represents using IPsec transport mode protection for the tunnel specified by parent tunnel encapsulation TLV, its value is a IPsec configuration tag as defined in [I-D.hujun-idr-bgp-ipsec].

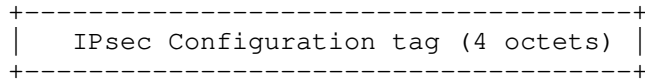


Figure 2: IPsec Configuration Tag

For a given tunnel encapsulation TLV, IPsec configuration tag sub-TLV MUST appear only one time.

2. Semantics and Operation

Except for what this document explicitly specifies, the semantics and operation of tunnel encapsulation TLV with IPsec Transport Protected sub-TLV are same as defined in [I-D.ietf-idr-tunnel-encaps] and [I-D.hujun-idr-bgp-ipsec].

IPsec Transport Protected sub-TLV MAY be included in any type of IP tunnel TLV specified in [I-D.ietf-idr-tunnel-encaps]; it MUST be ignored when included in a IPsec tunnel TLV.

The inclusion of IPsec Transport Protected TLV and its value is determined by local policy.

Following are the rules of operations:

1. All routers are pre-provisioned with Mapping between IPsec configuration tag value and IPsec configurations include authentication method/credentials
2. If a given NLRI needs a specific tunnel encapsulation with IPsec transport mode protection, then advertising router need to include an IPsec Transport Protected sub-TLV with required configuration tag, in the corresponding tunnel encapsulation TLV/attribute, along with the NLRI in BGP UPDATE U;
3. When a router need to forward a packet along a path is determined by a BGP UPDATE which has a tunnel encapsulation attribute that contains one or more tunnel TLV, router selects a tunnel TLV based on Semantics defined in [I-D.ietf-idr-tunnel-encaps], if

the selected tunnel TLV contains IPsec Transport Protected sub-TLV, then the router use first feasible CHILD_SA for IP tunnel packet encryption, a CHILD SA is considered as feasible when it meets all following conditions:

- * it is ESP transport mode
 - * its private and public routing instance is same as routing instance in which the packet to be forwarded
 - * its peer tunnel address is same as indicated by Remote Endpoint sub-TLV
 - * the source and destination address of the packet to be forwarded falls in the range of CHILD SA's traffic selector
 - * its transform and other configuration maps to the tag indicated in the IPsec configuration tag sub-TLV
4. If router can't find such CHILD SA, then it will use IKEv2 to create one with following IPsec configuration:
- * ESP transport mode
 - * private and public routing instance is the routing instance in which the packet to be forwarded
 - * peer tunnel address is specified by Remote Endpoint sub-TLV
 - * local traffic selector:
 - + address range: local tunnel endpoint address
 - + protocol: tag mapped configuration
 - + port range: tag mapped configuration
 - * remote traffic selector:
 - + address range: address in Remote Endpoint sub-TLV of selected tunnel encapsulation TLV
 - + protocol: tag mapped configuration
 - + port range: tag mapped configuration

- * other configurations come from mapping of the configuration tag in IPsec Transport Protected sub-TLV of selected tunnel encapsulation TLV

3. IANA Considerations

This document will request new values in IANA "BGP Tunnel Encapsulation Attribute Sub-TLVs" registry for IPsec Transport Protected sub-TLV.

4. Security Considerations

IKEv2 is used to create IPsec tunnel, which ensures following:

- o Traffic protection keys are generated dynamically during IKEv2 negotiation, only known by participating peer of the IPsec tunnel; there is no central node to manage and distribute all keys.
- o IKEv2 rekey mechanism refresh keys regularly; PFS(Perfect Forward Secrecy) provides additional protection;
- o Secure authentication mechanism that only allow authenticated peer to create tunnel
- o Traffic Selector guarantee that only agreed traffic is allowed to be forwarded within the IPsec tunnel;
- o Using a separate, dedicate protocol(IKEv2) for key management/authentication ensure they are not tied to BGP, all existing and future IKEv2 features could be used without changing BGP;

There is concern that malicious party might manipulate IPsec tunnel encapsulation attribute to divert traffic, however this risk could be mitigated by IKEv2 mutual authentication.

BGP route filter include outbound route filter [RFC5291], Origin Validation [RFC6811] and BGPSec [RFC8205] could be used to further secure BGP UPDATE message.

IKEv2 cookie [RFC7296] and varies mechanisms defined including client puzzle defined in [RFC8019] could be used to protect IKEv2 from Distributed Denial-of-Service Attacks.

Follow latest IETF ESP/IKEv2 implementation requirement and guidance ([RFC8221] and [RFC8247] at time of writing) to make sure always using secure and up-to-date cryptographic algorithms;

5. Change Log

- o v00 Sep 29, 2019: initial draft

6. References

6.1. Normative References

- [I-D.hujun-idr-bgp-ipsec]
Hu, J., "BGP Provisioned IPsec Tunnel Configuration",
draft-hujun-idr-bgp-ipsec-01 (work in progress), September
2019.
- [I-D.ietf-idr-tunnel-encaps]
Patel, K., Velde, G., and S. Ramachandra, "The BGP Tunnel
Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-14
(work in progress), September 2019.
- [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>>.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the
Internet Protocol", RFC 4301, DOI 10.17487/RFC4301,
December 2005, <<https://www.rfc-editor.org/info/rfc4301>>.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)",
RFC 4303, DOI 10.17487/RFC4303, December 2005,
<<https://www.rfc-editor.org/info/rfc4303>>.
- [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>>.

6.2. Informative References

- [RFC5291] Chen, E. and Y. Rekhter, "Outbound Route Filtering
Capability for BGP-4", RFC 5291, DOI 10.17487/RFC5291,
August 2008, <<https://www.rfc-editor.org/info/rfc5291>>.
- [RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R.
Austein, "BGP Prefix Origin Validation", RFC 6811,
DOI 10.17487/RFC6811, January 2013,
<<https://www.rfc-editor.org/info/rfc6811>>.

- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <<https://www.rfc-editor.org/info/rfc7296>>.
- [RFC8019] Nir, Y. and V. Smyslov, "Protecting Internet Key Exchange Protocol Version 2 (IKEv2) Implementations from Distributed Denial-of-Service Attacks", RFC 8019, DOI 10.17487/RFC8019, November 2016, <<https://www.rfc-editor.org/info/rfc8019>>.
- [RFC8205] Lepinski, M., Ed. and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <<https://www.rfc-editor.org/info/rfc8205>>.
- [RFC8221] Wouters, P., Migault, D., Mattsson, J., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 8221, DOI 10.17487/RFC8221, October 2017, <<https://www.rfc-editor.org/info/rfc8221>>.
- [RFC8247] Nir, Y., Kivinen, T., Wouters, P., and D. Migault, "Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 8247, DOI 10.17487/RFC8247, September 2017, <<https://www.rfc-editor.org/info/rfc8247>>.

Author's Address

Hu Jun
Nokia
777 East Middlefield Road
Mountain View CA 95148
United States

Email: jun.hu@nokia.com

Inter-Domain Routing
Internet-Draft
Intended status: Standards Track
Expires: November 14, 2022

K. Talaulikar, Ed.
Arrcus Inc
P. Psenak
Cisco Systems
J. Tantsura
Microsoft
May 13, 2022

Application-Specific Attributes Advertisement with BGP Link-State
draft-ietf-idr-bgp-ls-app-specific-attr-11

Abstract

New extensions have been defined for link-state routing protocols that enable distribution of application-specific link attributes for existing as well as newer applications such as Segment Routing. This document defines extensions to BGP-LS to enable the advertisement of these application-specific attributes as a part of the topology information from the network.

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 November 14, 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
2. Application Specific Link Attributes TLV	3
3. Application-Specific Link Attributes	4
4. Procedures	5
5. Deployment Considerations	8
6. IANA Considerations	8
7. Manageability Considerations	9
8. Security Considerations	9
9. Acknowledgements	9
10. References	10
10.1. Normative References	10
10.2. Informative References	10
Authors' Addresses	11

1. Introduction

BGP Link-State [RFC7752] enables the distribution of the link-state topology information from link-state routing protocols (viz., IS-IS [RFC1195], OSPFv2 [RFC2328] and OSPFv3 [RFC5340]) to an application like a controller or Path Computation Engine (PCE) via BGP. The controller/PCE gets the end-to-end topology information across IGP domains so it can perform path computations for use cases like end-to-end traffic engineering (TE).

The link-state topology information distributed via BGP-LS includes link attributes that were originally defined for traditional MPLS Traffic Engineering (i.e., using RSVP-TE [RFC3209]) or GMPLS [RFC4202]) applications. In recent years new applications, such as Segment Routing (SR) Policy [RFC8402] and Loop-Free Alternates (LFA) [RFC5286], that also make use of link attributes have been introduced. [RFC8919] and [RFC8920] define extensions for IS-IS and OSPF respectively that enable advertising application-specific link attributes for these and other future applications. This has resulted in the need for a similar BGP-LS extension to include this additional link-state topology information from the link-state routing protocols.

This document defines the BGP-LS extensions for the advertisement of application-specific link attributes. It describes the advertisement of these link attributes as top-level TLVs (i.e., as TLVs of the BGP-

LS Attribute) and as sub-TLVs of the new (top-level) Application Specific Link Attributes TLV. The document also describes the procedures for the advertisement of these attributes from the underlying IGP and discusses their deployment aspects.

1.1. Requirements Language

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

2. Application Specific Link Attributes TLV

The BGP-LS [RFC7752] specifies the Link Network Layer Reachability Information (NLRI) for the advertisement of links and their attributes using the BGP-LS Attribute. The Application-Specific Link Attributes (ASLA) TLV is a new optional top-level BGP-LS Attribute TLV that is introduced for Link NLRIs. It is defined such that it may act as a container for certain existing and future link attributes that require application-specific definition.

The format of this TLV is as follows and is similar to the corresponding ASLA sub-TLVs defined for OSPF and IS-IS in [RFC8920] and [RFC8919] respectively.

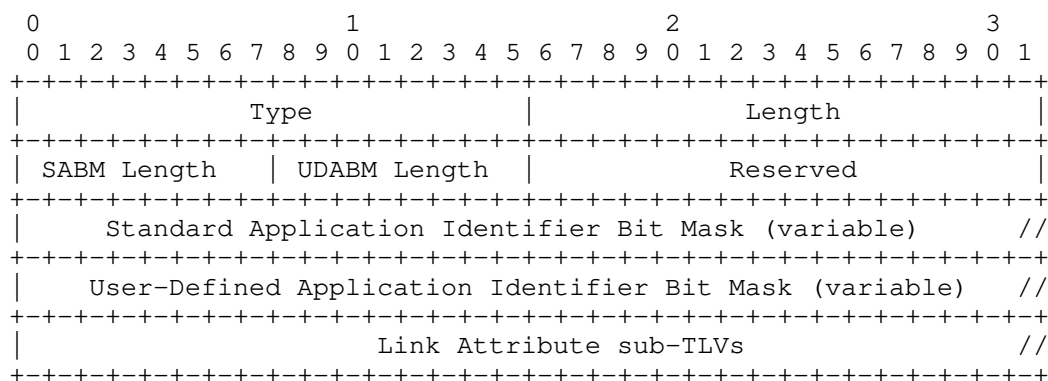


Figure 1: Application-Specific Link Attributes TLV

where:

- o Type: 1122
- o Length: variable.

- o SABM Length : Standard Application Identifier Bit Mask Length in octets as defined in [RFC8920].
- o UDABM Length : User-Defined Application Identifier Bit Mask Length in octets as defined in [RFC8920].
- o Standard Application Identifier Bit Mask : An optional set of bits (of size 0, 4, or 8 octets as indicated by the SABML), where each bit represents a single standard application as defined in [RFC8919].
- o User-Defined Application Identifier Bit Mask : An optional set of bits (of size 0, 4, or 8 octets as indicated by the UDABML), where each bit represents a single user-defined application as defined in [RFC8919].
- o Link Attribute sub-TLVs : BGP-LS Attribute TLVs corresponding to the Link NLRI that are application-specific (including existing ones as specified in Section 3) are included as sub-TLVs of the ASLA TLV.

The semantics associated with the standard and user-defined bit masks as well as the encoding scheme for application-specific attributes are as specified in [RFC8920].

The ASLA TLV and its sub-TLVs can only be added to the BGP-LS Attribute associated with the Link NLRI of the node that originates the underlying IGP link attribute TLVs/sub-TLVs. The procedures for originating link attributes in the ASLA TLV from underlying IGPs are specified in Section 4.

3. Application-Specific Link Attributes

Several BGP-LS Attribute TLVs corresponding to the Link NLRI are defined in BGP-LS and more may be added in the future. Those attributes that have been determined to be, and advertised as application-specific in the underlying IGPs are also encoded in a similar manner in BGP-LS.

The following table lists the currently defined BGP-LS Attributes TLVs corresponding to Link NLRI that can have application-specific semantics based on the underlying IGP specifications [RFC8919] [RFC8920]. These were originally defined with semantics for RSVP-TE and GMPLS applications in BGP-LS by the respective reference documents.

TLV Code Point	Description	Reference Document
1088	Administrative group (color)	[RFC7752]
1092	TE Default Metric	[RFC7752]
1096	Shared Risk Link Group	[RFC7752]
1114	Unidirectional Link Delay	[RFC8571]
1115	Min/Max Unidirectional Link Delay	[RFC8571]
1116	Unidirectional Delay Variation	[RFC8571]
1117	Unidirectional Link Loss	[RFC8571]
1118	Unidirectional Residual Bandwidth	[RFC8571]
1119	Unidirectional Available Bandwidth	[RFC8571]
1120	Unidirectional Utilized Bandwidth	[RFC8571]
1173	Extended Administrative Group	[RFC9104]

Table 1: Existing BGP-LS TLVs identified as Application-Specific

All the BGP-LS Attribute TLVs defined in the table above are REQUIRED to be advertised as a top-level TLV in the BGP-LS Attribute for carrying link attributes specific to RSVP-TE.

BGP-LS Attribute TLVs corresponding to Link NLRI that are identified as application-specific are REQUIRED to be encoded within an ASLA TLV.

Link attributes that do not have application-specific semantics MUST NOT be advertised within the ASLA TLV.

When the same application-specific link attributes are advertised both within the ASLA TLV and as top-level TLVs in the BGP-LS Attribute, the attributes advertised within the ASLA TLV take precedence for the applications indicated in the ASLA TLV encoding.

4. Procedures

The BGP-LS originator learns of the association of an application-specific attribute to one or more applications from the underlying IGP protocol LSA/LSPs from which it is advertising the topology information. [RFC8920] and [RFC8919] specify the mechanisms for advertising application-specific link attributes in OSPF and IS-IS respectively.

Application-specific link attributes received from an IGP node without the use of ASLA encodings continue to be encoded using the respective BGP-LS top-level TLVs listed in Table 1 as specified in their respective reference documents.

A BGP-LS originator node that is advertising link-state information from the underlying IGP using ASLA encodings determines their BGP-LS encoding based on the following rules:

1. Application-specific link attributes received from an OSPF node using ASLA sub-TLV or from an IS-IS node using either ASLA sub-TLV or ASLA SRLG TLV MUST be encoded in the BGP-LS ASLA TLV as sub-TLVs. Exceptions to this rule are specified in (2)(F) and (2)(G) below.
2. In the case of IS-IS, the following specific procedures are to be followed:
 - A. When application-specific link attributes are received from a node with the L bit set in the ASLA sub-TLV and application bits other than RSVP-TE are set in the application bit masks then the application-specific link attributes advertised in the corresponding legacy IS-IS TLVs/sub-TLVs MUST be encoded within the BGP-LS ASLA TLV as sub-TLVs with the application bits, other than the RSVP-TE bit, copied from the IS-IS ASLA sub-TLV. The link attributes advertised in the legacy IS-IS TLVs/sub-TLVs are also advertised in BGP-LS top-level TLVs as per [RFC7752] [RFC8571] [RFC9104]. The same procedure also applies for the advertisement of the SRLG values from the IS-IS ASLA SRLG TLV.
 - B. When the ASLA sub-TLV has the RSVP-TE application bit set, then the link attributes for the corresponding ASLA sub-TLV MUST be encoded using the respective BGP-LS top-level TLVs as per [RFC7752] [RFC8571] [RFC9104]. Similarly, when the ASLA SRLG TLV has the RSVP-TE application bit set, then the SRLG values within it MUST be encoded using the top-level BGP-LS SRLG TLV (1096) as per [RFC7752].
 - C. The SRLGs advertised in IS-IS SRLG ASLA TLVs and the other link attributes advertised in IS-IS ASLA sub-TLVs are REQUIRED to be collated, on a per-application basis, for all applications that have their bit set in the SABM/UDABM in at least one of the aforementioned TLV types. When performing this collation, only the TLVs with the application's bit set in SABM/UDABM MUST be used when such TLVs are available from either TLV types. If the bit for an application is set in the SABM/UDABM of only one of the TLV types, then the

attributes from the other TLV type with zero-length application bit mask MUST be also collated for that application, if such TLV is available. Such collated link attributes are advertised in a per-application instance of the BGP-LS ASLA TLV.

- D. If the resulting set of collated link attributes and SRLG values is common across multiple applications, they MAY be advertised in a common BGP-LS ASLA TLV instance where the bits for all such applications would be set in the application bit mask.
- E. Both the SRLG values from IS-IS ASLA SRLG TLVs and the link attributes from IS-IS ASLA sub-TLVs, with the zero-length application bit mask, MUST be advertised into a BGP-LS ASLA TLV with a zero-length application bit mask, independent of the collation described above.
- F. [RFC8919] allows the advertisement of the Maximum Link Bandwidth within an ASLA sub-TLV even though it is not an application-specific attribute. However, when originating the Maximum Link Bandwidth into BGP-LS, the attribute MUST be encoded only in the top-level BGP-LS Maximum Link Bandwidth TLV (1089) and MUST NOT be advertised within the BGP-LS ASLA TLV.
- G. [RFC8919] also allows the advertisement of the Maximum Reservable Link Bandwidth and the Unreserved Bandwidth within an ASLA sub-TLV even though these attributes are specific to RSVP-TE application. However, when originating the Maximum Reservable Link Bandwidth and Unreserved Bandwidth into BGP-LS, these attributes MUST be encoded only in the BGP-LS top-level Maximum Reservable Link Bandwidth TLV (1090) and Unreserved Bandwidth TLV (1091) respectively and not within the BGP-LS ASLA TLV.

These rules ensure that a BGP-LS originator performs the advertisement for all application-specific link attributes from the IGP nodes that support or do not support the ASLA extension. Furthermore, it also ensures that the top-level BGP-LS TLVs defined for RSVP-TE and GMPLS applications continue to be used for advertisement of their application-specific attributes.

A BGP-LS speaker would normally advertise all the application-specific link attributes corresponding to RSVP-TE and GMPLS applications as existing top-level BGP-LS TLVs while for other applications they are encoded in ASLA TLV(s) with appropriate applicable bit mask setting. The application-specific attribute

value received via sub-TLVs within the ASLA TLV have preference over the value received via the top-level TLVs.

5. Deployment Considerations

BGP-LS sources the link-state topology information (including the extensions introduced by this document) from the underlying link-state IGP protocols. The various deployment aspects related to the advertisement and use of application-specific link attributes are discussed in the Deployment Considerations sections of [RFC8920] and [RFC8919]. The IGP backward compatibility aspects described in those documents associated with application-specific link attributes along with the BGP-LS procedures specified in this document enable backward compatibility in deployments of existing implementations of [RFC7752] [RFC8571] [RFC9104] for applications such as RSVP-TE, SR Policy, and LFA.

It is recommended that nodes supporting this specification are selected as originators of BGP-LS information when advertising the link-state information from the IGPs.

BGP-LS consumers that do not support this specification can continue to use the existing top-level TLVs for link attributes for existing applications as discussed above. They would, however, not be able to support neither the application-specific link attributes nor newer applications that may be encoded only using the ASLA TLV.

6. IANA Considerations

IANA has assigned, through the early allocation process, the following code-point from the "BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs" registry. This document requests that the allocation be made permanent. The column "IS-IS TLV/Sub-TLV" defined in the registry does not require any value and should be left empty.

Code Point	Description	Reference
1122	Application-Specific Link Attributes	this document

Table 2: ASLA TLV Code-Point Allocation

7. Manageability Considerations

The new protocol extensions introduced in this document augment the existing IGP topology information defined in [RFC7752]. Procedures and protocol extensions defined in this document do not affect the BGP protocol operations and management other than as discussed in the Manageability Considerations section of [RFC7752]. Specifically, the malformed NLRIs attribute tests in the Fault Management section of [RFC7752] now encompasses the BGP-LS TLVs defined in this document.

The extensions specified in this document do not specify any new configuration or monitoring aspects in BGP or BGP-LS. The specification of BGP models is an ongoing work based on [I-D.ietf-idr-bgp-model].

8. Security Considerations

Security considerations for acquiring and distributing BGP-LS information are discussed in [RFC7752].

The TLVs introduced in this document are used to propagate the application-specific link attributes IGP extensions defined in [RFC8919] and [RFC8920]. It is assumed that the IGP instances originating these TLVs will support all the required security (as described in [RFC8919] and [RFC8920]) to prevent any security issues when propagating the TLVs into BGP-LS.

This document defines a new way to advertise link attributes. Tampering with the information defined in this document may affect applications using it, including impacting traffic engineering, which uses various link attributes for its path computation. As the advertisements defined in this document limit the scope to specific applications, the impact of tampering is similarly limited in scope. The advertisement of the link attribute information defined in this document presents no significant additional risk beyond that associated with the existing link attribute information already supported in [RFC7752].

9. Acknowledgements

The authors would like to thank Les Ginsberg, Baalajee S, Amalesh Maity, Acee Lindem, Keyur Patel, Paul Wouters, Rudy Selderslaghs, and Kristy Paine for their review and feedback on this document. The authors would also like to thank Alvaro Retana for his very detailed AD review and comments for improving this document.

10. References

10.1. Normative References

- [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>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<https://www.rfc-editor.org/info/rfc7752>>.
- [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>>.
- [RFC8571] Ginsberg, L., Ed., Previdi, S., Wu, Q., Tantsura, J., and C. Filsfils, "BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions", RFC 8571, DOI 10.17487/RFC8571, March 2019, <<https://www.rfc-editor.org/info/rfc8571>>.
- [RFC8919] Ginsberg, L., Psenak, P., Previdi, S., Henderickx, W., and J. Drake, "IS-IS Application-Specific Link Attributes", RFC 8919, DOI 10.17487/RFC8919, October 2020, <<https://www.rfc-editor.org/info/rfc8919>>.
- [RFC8920] Psenak, P., Ed., Ginsberg, L., Henderickx, W., Tantsura, J., and J. Drake, "OSPF Application-Specific Link Attributes", RFC 8920, DOI 10.17487/RFC8920, October 2020, <<https://www.rfc-editor.org/info/rfc8920>>.
- [RFC9104] Tantsura, J., Wang, Z., Wu, Q., and K. Talaulikar, "Distribution of Traffic Engineering Extended Administrative Groups Using the Border Gateway Protocol - Link State (BGP-LS)", RFC 9104, DOI 10.17487/RFC9104, August 2021, <<https://www.rfc-editor.org/info/rfc9104>>.

10.2. Informative References

- [I-D.ietf-idr-bgp-model] Jethanandani, M., Patel, K., Hares, S., and J. Haas, "BGP YANG Model for Service Provider Networks", draft-ietf-idr-bgp-model-13 (work in progress), March 2022.

- [RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, DOI 10.17487/RFC1195, December 1990, <<https://www.rfc-editor.org/info/rfc1195>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <<https://www.rfc-editor.org/info/rfc3209>>.
- [RFC4202] Kompella, K., Ed. and Y. Rekhter, Ed., "Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4202, DOI 10.17487/RFC4202, October 2005, <<https://www.rfc-editor.org/info/rfc4202>>.
- [RFC5286] Atlas, A., Ed. and A. Zinin, Ed., "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, DOI 10.17487/RFC5286, September 2008, <<https://www.rfc-editor.org/info/rfc5286>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <<https://www.rfc-editor.org/info/rfc5340>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.

Authors' Addresses

Ketan Talaulikar (editor)
Arrcus Inc
India

Email: ketant.ietf@gmail.com

Peter Psenak
Cisco Systems
Slovakia

Email: ppsenak@cisco.com

Jeff Tantsura
Microsoft

Email: jefftant.ietf@gmail.com

Interdomain Routing
Internet-Draft
Intended status: Standards Track
Expires: 7 September 2022

M. Jethanandani
Kloud Services
K. Patel
Arrcus
S. Hares
Huawei
J. Haas
Juniper Networks
6 March 2022

BGP YANG Model for Service Provider Networks
draft-ietf-idr-bgp-model-13

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on 7 September 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components

extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
1.1. Goals and approach	3
1.2. Note to RFC Editor	4
1.3. Terminology	5
1.4. Abbreviations	5
2. Model overview	5
2.1. BGP protocol configuration	6
2.2. Policy configuration overview	9
2.3. BGP RIB overview	9
2.3.1. Local Routing	11
2.3.2. Pre updates per-neighbor	11
2.3.3. Post updates per-neighbor	11
2.3.4. Pre route advertisements per-neighbor	11
2.3.5. Post route advertisements per-neighbor	11
3. Relation to other YANG data models	11
4. Security Considerations	12
5. IANA Considerations	13
5.1. URI Registration	13
5.2. YANG Module Name Registration	14
6. YANG modules	14
7. Structure of the YANG modules	15
7.1. Main module and submodules for base items	15
7.2. BGP types	66
7.3. BGP policy data	79
7.4. RIB modules	94
8. Contributors	124
9. Acknowledgements	124
10. References	124
10.1. Normative references	124
10.2. Informative references	128
Appendix A. Examples	129
A.1. Creating BGP Instance	129
A.2. Neighbor Address Family Configuration	130
A.3. IPv6 Neighbor Configuration	131
A.4. VRF Configuration	132
A.5. BGP Policy	134
Appendix B. How to add a new AFI and Augment a Module	138
Appendix C. How to deviate a module	142
Appendix D. Complete configuration tree diagram	142
Appendix E. Complete policy tree diagram	163
Authors' Addresses	165

1. Introduction

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

1.1. Goals and approach

The model covers the base BGP features that are deployed across major implementations and the common BGP configurations in use across a number of operator network deployments. In particular, this model attempts to cover BGP features defined in BGP [RFC4271], BGP Communities Attribute [RFC1997], BGP Route Reflection [RFC4456], Multiprotocol Extensions for BGP-4 [RFC4760], Autonomous System Confederations for BGP [RFC5065], BGP Route Flap Damping [RFC2439], Graceful Restart Mechanism for BGP [RFC4724], BGP Prefix Origin Validation [RFC6811], and Advertisement of Multiple Paths in BGP [RFC7911].

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

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

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

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

- * The address families that are supported by peers, and the global configuration which relates to them.
- * The policy configuration "hooks" and BGP-specific policy features that relate to a neighbor - controlling the import and export of NLRI's.
- * BGP RIB contents.

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

Where possible, naming in the model follows conventions used in available standards documents, and otherwise tries to be self-explanatory with sufficient descriptions of the intended behavior. Similarly, configuration data value constraints and default values, where used, are based on recommendations in current standards documentation, or those commonly used in multiple implementations. Since implementations can vary widely in this respect, this version of the model specifies only a limited set of defaults and ranges with the expectation of being more prescriptive in future versions based on actual operator use.

1.2. Note to RFC Editor

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

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

2022-03-06 with the actual date of the publication of this document.

RFC ZZZZ, where ZZZZ is the number assigned to A YANG Data Model for Routing Policy Management [RFC9067].

1.3. Terminology

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

1.4. Abbreviations

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

Table 1

2. Model overview

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

- * base protocol configuration -- configuration affecting BGP protocol-related operations, defined at various levels of hierarchy.
- * multiprotocol configuration -- configuration affecting individual address-families within BGP Multiprotocol Extensions for BGP-4 [RFC4760].
- * neighbor configuration -- configuration affecting an individual neighbor within BGP.
- * neighbor multiprotocol configuration -- configuration affecting individual address-families for a neighbor within BGP.

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

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

2.1. BGP protocol configuration

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

```
module: ietf-bgp
```

```
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol:
    +--rw bgp
      +--rw global!
        +--rw as inet:as-number
        +--rw identifier? yang:dotted-quad
        +--rw distance
        |   ...
        +--rw confederation
        |   ...
        +--rw graceful-restart {bt:graceful-restart}?
        |   ...
        +--rw use-multiple-paths
        |   ...
        +--rw route-selection-options
        |   ...
        +--rw afi-safis
        |   ...
        +--rw apply-policy
        |   ...
        +--ro total-paths? uint32
        +--ro total-prefixes? uint32
      +--rw neighbors
        +--rw neighbor* [remote-address]
        |   ...
        +---n established
        |   ...
```

```

|   +---n backward-transition
|   |   ...
|   +---x clear {bt:clear-neighbors}?
|   |   ...
+--rw peer-groups
|   +--rw peer-group* [name]
|   |   ...
+--rw interfaces
|   +--rw interface* [name]
|   |   ...
+--ro rib
|   +--ro attr-sets
|   |   ...
|   +--ro communities
|   |   ...
|   +--ro ext-communities
|   |   ...
|   +--ro large-communities
|   |   ...
|   +--ro afi-safis
|   |   ...

```

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

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

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

Address-family configuration is made available in multiple points within the model - primarily within the global container, where instance-wide configuration can be set (for example, global protocol

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

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

```

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

```

2.2. Policy configuration overview

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

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

- * within the global instance, where a policy applies to all address-families for all peers.
- * on a global AFI-SAFI basis, where policies apply to all peers for a particular address-family.
- * on a per-peer-group or per-neighbor basis - where the policy applies to all address-families for the particular group or neighbor.
- * on a per-afi-safi basis within a neighbor or peer-group context, where the policy is specific to the AFI-SAFI for a a specific neighbor or group.

module: ietf-bgp-policy

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

2.3. BGP RIB overview

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

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

```

module: ietf-bgp

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

```

2.3.1. Local Routing

The loc-rib is the main BGP routing table for the local routing instance, containing best-path selections for each prefix. The loc-rib table may contain multiple routes for a given prefix, with an attribute to indicate which was selected as the best-path. Note that multiple paths may be used or advertised even if only one path is marked as best, e.g., when using BGP add-paths. An implementation may choose to mark multiple paths in the RIB as best-path by setting the flag to true for multiple entries.

2.3.2. Pre updates per-neighbor

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

2.3.3. Post updates per-neighbor

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

2.3.4. Pre route advertisements per-neighbor

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

2.3.5. Post route advertisements per-neighbor

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

3. Relation to other YANG data models

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

4. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446]. The NETCONF Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

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

- The attribute 'as'. If a user is allowed to change this attribute, it will have the net effect of bringing down the entire routing instance, causing it to delete all the current routing entries, and learning new ones.
- The attribute 'identifier'. If a user is allowed to change this attribute, it will have the net effect of this routing instance re-advertising all its routes.
- The attribute 'distance'. If a user is allowed to change this attribute, it will cause the preference for routes, e.g. external vs internal to change.
- The attribute 'enabled' in the 'confederation' container. This attribute defines whether a local-AS is part of a BGP federation.
- Finally, there are a whole set of route selection options such as 'always-compare-med', 'ignore-as-path-length' that affect the way the system picks up a particular route. Being able to change will adversely affect how the route selection happens.

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

- The list of neighbors, and their attributes. Allowing a user to read these attributes, in particular the address/port information may allow a malicious user to launch an attack at the particular address/port.
- The 'rib' container. This container contains sensitive information such as attribute sets, communities and external communities. Being able to read the contents of this container will allow a malicious user to understand how the system decide how to route a packet, and thus try to affect a change.

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

- The model allows for routes to be cleared using the 'clear' RPC operations, causing the entire RIB table to be cleared.
- The model allows for statistics to be cleared by the 'clear' RPC operation, causing all the individual statistics to be cleared.
- The model also allows for neighbors that have been learnt by the system to be cleared by using the 'clear' RPC operation.

BGP OPSEC [RFC7454] describes several policies that can be used to secure a BGP. In particular, it recommends securing the underlying TCP session and to use Generalized TTL Security Mechanism (GTSM) [RFC5082] capability to make it harder to spoof a BGP session. This module allows implementations that want to support the capability to configure a TTL value, under a feature flag. It also defines a container 'secure-session' that can be augmented with TCP-Authentication Option (TCP-AO) [RFC5925], or other methods to secure a BGP session, and will be developed in a future version of this draft.

5. IANA Considerations

This document registers three URIs and three YANG modules.

5.1. URI Registration

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

URI: urn:ietf:params:xml:ns:yang:ietf-bgp
URI: urn:ietf:params:xml:ns:yang:ietf-bgp-policy
URI: urn:ietf:params:xml:ns:yang:ietf-bgp-types

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

5.2. YANG Module Name Registration

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

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

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

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

6. YANG modules

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

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

- * `ietf-bgp-common` - defines the groupings that are common across more than one context (where contexts are neighbor, group, global)
- * `ietf-bgp-common-multiprotocol` - defines the groupings that are common across more than one context, and relate to multiprotocol BGP
- * `ietf-bgp-common-structure` - defines groupings that are shared by multiple contexts, but are used only to create structural elements, i.e., containers (leaf nodes are defined in separate groupings)
- * `ietf-bgp-neighbor` - groupings with data specific to the neighbor context
- * `ietf-bgp-rib` - grouping for representing BGP RIB.

Additionally, modules include:

- * ietf-bgp-types - common type and identity definitions for BGP, including BGP policy
- * ietf-bgp-policy - BGP-specific policy data definitions for use with [RFC9067] (described in more detail Section 2.2)

7. Structure of the YANG modules

The YANG model can be subdivided between the main module for base items, types, policy data, and the RIB module. It references BGP Communities Attribute [RFC1997], Route Refresh Capability for BGP-4 [RFC2918], NOPEER Community for BGP [RFC3765], BGP/MPLS IP Virtual Private Networks (VPNs) [RFC4364], BGP MED Considerations [RFC4451], BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN [RFC4659], Graceful Restart Mechanism for BGP [RFC4724], Multiprotocol Extensions for BGP-4 [RFC4760], Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling [RFC4761], Autonomous System Configuration for BGP [RFC5065], The Generalized TTL Security Mechanism (GTSM) [RFC5082], Bidirectional Forward Detection (BFD) [RFC5880], Bidirectional Forward Detection for IPv4 and IPv6 (Single Hop) [RFC5881], Bidirectional Forwarding Detection (BFD) for Multihop Paths [RFC5883], The TCP Authentication Option [RFC5925], BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs [RFC6514], BGP Support for Four-Octet Autonomous System (AS) Number Space [RFC6793], Advertisement of Multiple Paths in BGP [RFC7911], YANG Key Chain [RFC8177], Carrying Label Information in BGP-4 [RFC8277], A YANG Data Model for Routing Policy [RFC9067], YANG Data Model for Bidirectional Forward Detection [RFC9127], and YANG Model for Transmission Control Protocol (TCP) Configuration [I-D.ietf-tcpm-yang-tcp].

7.1. Main module and submodules for base items

```
<CODE BEGINS> file "ietf-bgp@2022-03-06.yang"
module ietf-bgp {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-bgp";
  prefix bgp;

  /*
   * Import and Include
   */

  import ietf-routing {
    prefix rt;
    reference
```

```
    "RFC 8349, A YANG Data Model for Routing Management
      (NMDA Version).";
  }
  import ietf-routing-policy {
    prefix rt-pol;
    reference
      "RFC ZZZZ, A YANG Data Model for Routing Policy Management.";
  }
  import ietf-interfaces {
    prefix if;
    reference
      "RFC 8343, A YANG Data Model for Interface Management.";
  }
  import ietf-bgp-types {
    prefix bt;
    reference
      "RFC XXXX, BGP YANG Model for Service Provider Network.";
  }
  import ietf-bfd-types {
    prefix bfd-types;
    reference
      "I-D.ietf-bfd-rfc9127-bis: YANG Data Model for
        Bidirectional Forward Detection (BFD).";
  }
  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types.";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types.";
  }
  import ietf-key-chain {
    prefix key-chain;
    reference
      "RFC 8177: YANG Key Chain.";
  }
  import ietf-tcp {
    prefix tcp;
    reference
      "I-D.ietf-tcpm-yang-tcp: Transmission Control Protocol (TCP)
        YANG Model.";
  }
  include ietf-bgp-common {
    revision-date 2022-03-06;
  }
}
```

```
include ietf-bgp-common-multiprotocol {
  revision-date 2022-03-06;
}
include ietf-bgp-common-structure {
  revision-date 2022-03-06;
}
include ietf-bgp-neighbor {
  revision-date 2022-03-06;
}
include ietf-bgp-rib-types {
  revision-date 2022-03-06;
}
include ietf-bgp-rib {
  revision-date 2022-03-06;
}
include ietf-bgp-rib-attributes {
  revision-date 2022-03-06;
}
include ietf-bgp-rib-tables {
  revision-date 2022-03-06;
}
```

organization

"IETF IDR Working Group";

contact

"WG Web: <<http://tools.ietf.org/wg/idr>>
WG List: <idr@ietf.org>

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

description

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

This model supports the following BGP configuration level hierarchy:

```
BGP
|
```



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

Copyright (c) 2021 IETF Trust and the persons identified as authors of the code. All rights reserved.

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

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

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

```
revision 2022-03-06 {
  description
    "Initial Version";
  reference
    "RFC XXXX, BGP Model for Service Provider Network ";
}

/*
 * Identity
 */

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

/*
```

```
* Groupings
*/
grouping neighbor-and-peer-group-common {
  description
    "Neighbor and Peer Group configuration that is common.";

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

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

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

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

    leaf local-restarting {
      type boolean;
      config false;
      description
        "This flag indicates whether the local neighbor is
```

```
        currently restarting. The flag is cleared after all
        NLRI have been advertised to the peer, and the
        End-of-RIB (EOR) marker has been cleared.";
    }

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

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

/*
 * Containers
 */

augment "/rt:routing/rt:control-plane-protocols/"
```

```
+ "rt:control-plane-protocol" {
  when "derived-from-or-self(rt:type, 'bgp')" {
    description
      "This augmentation is valid for a routing protocol
       instance of BGP.";
  }
  description
    "BGP protocol augmentation of ietf-routing module
     control-plane-protocol.";
  container bgp {
    description
      "Top-level configuration for the BGP router.";
    container global {
      presence "Enables global configuration of BGP";
      description
        "Global configuration for the BGP router.";
      leaf as {
        type inet:as-number;
        mandatory true;
        description
          "Local autonomous system number of the router. Uses
           the 32-bit as-number type from the model in RFC 6991.";
      }
      leaf identifier {
        type yang:dotted-quad;
        description
          "BGP Identifier of the router - an unsigned 32-bit,
           non-zero integer that should be unique within an AS.
           The value of the BGP Identifier for a BGP speaker is
           determined upon startup and is the same for every local
           interface and BGP peer.";
        reference
          "RFC 6286: AS-Wide Unique BGP ID for BGP-4. Section 2.1";
      }
    }
    container distance {
      description
        "Administrative distances (or preferences) assigned to
         routes received from different sources (external, and
         internal).";
      leaf external {
        type uint8 {
          range "1..255";
        }
        description
          "Administrative distances for routes learned from
           external BGP (eBGP).";
      }
      leaf internal {
```

```
    type uint8 {
      range "1..255";
    }
    description
      "Administrative distances for routes learned from
      internal BGP (iBGP).";
  }
}
container confederation {
  description
    "Configuration options specifying parameters when the
    local router is within an autonomous system which is
    part of a BGP confederation.";
  leaf enabled {
    type boolean;
    description
      "When this leaf is set to true it indicates that
      the local-AS is part of a BGP confederation.";
  }
  leaf identifier {
    type inet:as-number;
    description
      "Confederation identifier for the autonomous system.";
  }
  leaf-list member-as {
    type inet:as-number;
    description
      "Remote autonomous systems that are to be treated
      as part of the local confederation.";
  }
}
container graceful-restart {
  if-feature "bt:graceful-restart";
  description
    "Parameters relating the graceful restart mechanism for
    BGP.";
  uses graceful-restart-config;
}
uses global-group-use-multiple-paths;
uses route-selection-options;
container afi-safis {
  description
    "List of address-families associated with the BGP
    instance.";
  list afi-safi {
    key "name";
    description
      "AFI,SAFI configuration available for the
```

```
        neighbor or group.";
    uses mp-afi-safi-config;
    uses state;
    container graceful-restart {
        if-feature "bt:graceful-restart";
        description
            "Parameters relating to BGP graceful-restart";
        uses mp-afi-safi-graceful-restart-config;
    }
    uses route-selection-options;
    uses global-group-use-multiple-paths;
    uses mp-all-afi-safi-list-contents;
}
}
uses rt-pol:apply-policy-group;
uses state;
}

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

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

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

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

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

```
leaf remote-port {
  type inet:port-number;
  config false;
  description
    "The remote port for the TCP connection
    between the BGP peers. Note that the
    objects local-addr, local-port, remote-addr, and
    reemote-port provide the appropriate
    reference to the standard MIB TCP
    connection table.";
}

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

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

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

leaf enabled {
  type boolean;
```

```
default "true";
description
  "Whether the BGP peer is enabled. In cases where the
   enabled leaf is set to false, the local system should
   not initiate connections to the neighbor, and should
   not respond to TCP connections attempts from the
   neighbor. If the state of the BGP session is
   ESTABLISHED at the time that this leaf is set to
   false, the BGP session should be ceased.

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

leaf secure-session-enable {
  type boolean;
  default "false";
  description
    "Does this session need to be secured?";
}

container secure-session {
  when "../secure-session-enable = 'true'";
  description
    "Container for describing how a particular BGP session
     is to be secured.";

  choice option {
    case ao {
      uses tcp:ao;
      leaf ao-keychain {
        type key-chain:key-chain-ref;
        description
          "Reference to the key chain that will be used by
           this model. Applicable for TCP-AO and TCP-MD5
           only";
        reference
          "RFC 8177: YANG Key Chain.";
      }
    }
  }
  description
```



```
        "Uses TCP-AO to secure the session. Parameters for
        those are defined as a grouping in the TCP YANG
        model.";
    reference
        "RFC 5925 - The TCP Authentication Option.";
}

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

description
    "Choice of authentication options.";
}
}

leaf ttl-security {
    if-feature "bt:ttl-security";
    type uint8;
    default "255";
    description
        "BGP Time To Live (TTL) security check.";
    reference
        "RFC 5082: The Generalized TTL Security Mechanism
        (GTSM),
        RFC 7454: BGP Operations and Security.";
}

uses neighbor-group-config;
uses neighbor-and-peer-group-common;

container afi-safis {
    description
        "Per-address-family configuration parameters associated
```

```
        with the neighbor";
    uses bgp-neighbor-afi-safi-list;
}

leaf session-state {
    type enumeration {
        enum idle {
            description
                "Neighbor is down, and in the Idle state of the
                FSM.";
        }
        enum connect {
            description
                "Neighbor is down, and the session is waiting for
                the underlying transport session to be
                established.";
        }
        enum active {
            description
                "Neighbor is down, and the local system is awaiting
                a connection from the remote peer.";
        }
        enum opensent {
            description
                "Neighbor is in the process of being established.
                The local system has sent an OPEN message.";
        }
        enum openconfirm {
            description
                "Neighbor is in the process of being established.
                The local system is awaiting a NOTIFICATION or
                KEEPALIVE message.";
        }
        enum established {
            description
                "Neighbor is up - the BGP session with the peer is
                established.";
        }
    }
    // notification does not like a non-config statement.
    // config false;
    description
        "The BGP peer connection state.";
    reference
        "RFC 4271, Section 8.1.2.";
}
leaf last-established {
    type yang:date-and-time;
```

```
    config false;
    description
        "This timestamp indicates the time that the BGP session
        last transitioned in or out of the Established state.
        The value is the timestamp in seconds relative to the
        Unix Epoch (Jan 1, 1970 00:00:00 UTC).

        The BGP session uptime can be computed by clients as
        the difference between this value and the current time
        in UTC (assuming the session is in the ESTABLISHED
        state, per the session-state leaf).";
}
leaf-list negotiated-capabilities {
    type identityref {
        base bt:bgp-capability;
    }
    config false;
    description
        "Negotiated BGP capabilities.";
}
leaf negotiated-hold-time {
    type uint16;
    config false;
    description
        "The negotiated hold-time for the BGP session";
}
leaf last-error {
    type binary {
        length "2";
    }
    config false;
    description
        "The last error code and subcode seen by this
        peer on this connection.  If no error has
        occurred, this field is zero.  Otherwise, the
        first byte of this two byte OCTET STRING
        contains the error code, and the second byte
        contains the subcode.";
    reference
        "RFC 4271, Section 4.5.";
}
leaf fsm-established-time {
    type yang:gauge32;
    units "seconds";
    config false;
    description
        "This timer indicates how long (in
        seconds) this peer has been in the
```

```
        established state or how long
        since this peer was last in the
        established state. It is set to zero when
        a new peer is configured or when the router is
        booted.";
    reference
        "RFC 4271, Section 8.";
}
leaf treat-as-withdraw {
    type boolean;
    default "false";
    description
        "Specify whether erroneous UPDATE messages for which
        the NLRI can be extracted are treated as though the
        NLRI is withdrawn - avoiding session reset";
    reference
        "RFC 7606: Revised Error Handling for BGP UPDATE
        Messages.";
}
leaf erroneous-update-messages {
    type uint32;
    config false;
    description
        "The number of BGP UPDATE messages for which the
        treat-as-withdraw mechanism has been applied based on
        erroneous message contents";
}

container bfd {
    if-feature "bt:bfd";
    uses bfd-types:client-cfg-parms;
    description
        "BFD configuration per-neighbor.";
}

container statistics {
    description
        "Statistics per neighbor.";

    leaf peer-fsm-established-transitions {
        type yang:counter64;
        config false;
        description
            "Number of transitions to the Established state for
            the neighbor session. This value is analogous to the
            bgpPeerFsmEstablishedTransitions object from the
            standard BGP-4 MIB";
        reference
```

```
        "RFC 4273, Definitions of Managed Objects for
        BGP-4.";
    }
    leaf fsm-established-transitions {
        type yang:counter32;
        config false;
        description
            "The total number of times the BGP FSM
            transitioned into the established state
            for this peer.";
        reference
            "RFC 4271, Section 8.";
    }
    container messages {
        config false;
        description
            "Counters for BGP messages sent and received from the
            neighbor";
        leaf in-total-messages {
            type yang:counter32;
            config false;
            description
                "The total number of messages received
                from the remote peer on this connection.";
            reference
                "RFC 4271, Section 4.";
        }
        leaf out-total-messages {
            type yang:counter32;
            config false;
            description
                "The total number of messages transmitted to
                the remote peer on this connection.";
            reference
                "RFC 4271, Section 4.";
        }
    }
    leaf in-update-elapsed-time {
        type yang:gauge32;
        units "seconds";
        config false;
        description
            "Elapsed time (in seconds) since the last BGP
            UPDATE message was received from the peer.
            Each time in-updates is incremented,
            the value of this object is set to zero (0).";
        reference
            "RFC 4271, Section 4.3.
            RFC 4271, Section 8.2.2, Established state.";
```

```
}
container sent {
  description
    "Counters relating to BGP messages sent to the
    neighbor";
  uses bgp-neighbor-counters-message-types-state;
}
container received {
  description
    "Counters for BGP messages received from the
    neighbor";
  uses bgp-neighbor-counters-message-types-state;
}
}
container queues {
  config false;
  description
    "Counters related to queued messages associated with
    the BGP neighbor";
  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";
  }
}
}
action clear {
  if-feature "bt:clear-statistics";
  description
    "Clear statistics action command.

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

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

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

notification established {
    leaf remote-address {
        type leafref {
            path "../../neighbor/remote-address";
        }
        description
            "IP address of the neighbor that went into established
            state.";
    }
    leaf last-error {
        type leafref {
            path "../../neighbor/last-error";
        }
        description
            "The last error code and subcode seen by this
            peer on this connection.  If no error has
            occurred, this field is zero.  Otherwise, the
            first octet of this two byte OCTET STRING
            contains the error code, and the second octet
            contains the subcode.";
        reference
            "RFC 4271, Section 4.5.";
    }
    leaf session-state {
        type leafref {
            path "../../neighbor/session-state";
        }
        description
            "The BGP peer connection state.";
        reference
            "RFC 4271, Section 8.2.2.";
    }
    description
        "The established event is generated
        when the BGP FSM enters the established state.";
}
```

```
notification backward-transition {
  leaf remote-addr {
    type leafref {
      path "../../neighbor/remote-address";
    }
    description
      "IP address of the neighbor that changed its state from
      established state.";
  }
  leaf last-error {
    type leafref {
      path "../../neighbor/last-error";
    }
    description
      "The last error code and subcode seen by this
      peer on this connection.  If no error has
      occurred, this field is zero.  Otherwise, the
      first byte of this two byte OCTET STRING
      contains the error code, and the second byte
      contains the subcode.";
    reference
      "RFC 4271, Section 4.5.";
  }
  leaf session-state {
    type leafref {
      path "../../neighbor/session-state";
    }
    description
      "The BGP peer connection state.";
    reference
      "RFC 4271, Section 8.2.2.";
  }
  description
    "The backward-transition event is
    generated when the BGP FSM moves from a higher
    numbered state to a lower numbered state.";
}

action clear {
  if-feature "bt:clear-neighbors";
  description
    "Clear neighbors action.";

  input {
    choice operation {
      default operation-admin;
      description
        "The type of operation for the clear action.";
      case operation-admin {
```



```
leaf admin {
  type empty;
  description
    "Closes the Established BGP session with a BGP
    NOTIFICATION message with the Administrative
    Reset error subcode.";
  reference
    "RFC 4486 - Subcodes for BGP Cease Notification
    Message.";
}
}
case operation-hard {
  leaf hard {
    type empty;
    description
      "Closes the Established BGP session with a BGP
      NOTIFICATION message with the Hard Reset error
      subcode.";
    reference
      "RFC 8538, Section 3 - Notification Message
      Support for BGP Graceful Restart.";
  }
}
case operation-soft {
  leaf soft {
    type empty;
    description
      "Re-sends the current Adj-Rib-Out to this
      neighbor.";
  }
}
case operation-soft-inbound {
  leaf soft-inbound {
    if-feature "bt:route-refresh";
    type empty;
    description
      "Requests the Adj-Rib-In for this neighbor to be
      re-sent using the BGP Route Refresh feature.";
  }
}
}

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

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

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

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

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

    leaf secure-session-enable {
      type boolean;
      default "false";
      description
        "Does this session need to be secured?";
    }

    container secure-session {
      when "../secure-session-enable = 'true'";
      description
        "Container for describing how a particular BGP session
          is to be secured.";

      choice option {
        case ao {
          uses tcp:ao;
          leaf ao-keychain {
            type key-chain:key-chain-ref;
            description
              "Reference to the key chain that will be used by
```

```
        this model. Applicable for TCP-AO and TCP-MD5
        only";
    reference
        "RFC 8177: YANG Key Chain.";
}
description
    "Uses TCP-AO to secure the session. Parameters for
    those are defined as a grouping in the TCP YANG
    model.";
reference
    "RFC 5925 - The TCP Authentication Option.";
}
case md5 {
    uses tcp:md5;
    leaf md5-keychain {
        type key-chain:key-chain-ref;
        description
            "Reference to the key chain that will be used by
            this model. Applicable for TCP-AO and TCP-MD5
            only";
        reference
            "RFC 8177: YANG Key Chain.";
    }
    description
        "Uses TCP-MD5 to secure the session. Parameters for
        those are defined as a grouping in the TCP YANG
        model.";
    reference
        "RFC 5925: The TCP Authentication Option.";
}
case ipsec {
    leaf sa {
        type string;
        description
            "Security Association (SA) name.";
    }
    description
        "Currently, the IPsec/IKE YANG model has no
        grouping defined that this model can use. When
        such a grouping is defined, this model can import
        the grouping to add the key parameters
        needed to kick off IKE.";
}
description
    "Choice of authentication options.";
}
}
```

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

uses neighbor-group-config;
uses neighbor-and-peer-group-common;

container afi-safis {
  description
    "Per-address-family configuration parameters
    associated with the peer-group.";
  list afi-safi {
    key "name";
    description
      "AFI, SAFI configuration available for the
      neighbor or group";
    uses mp-afi-safi-config;
    container graceful-restart {
      if-feature "bt:graceful-restart";
      description
        "Parameters relating to BGP graceful-restart";
      uses mp-afi-safi-graceful-restart-config;
    }
    uses bgp-neighbor-use-multiple-paths;
    uses mp-all-afi-safi-list-contents;
  }
}

container interfaces {
  list interface {
    key "name";
    leaf name {
      type if:interface-ref;
      description
        "Reference to the interface within the routing
        instance.";
    }
    container bfd {
```

```
        if-feature "bt:bfd";
        leaf enabled {
            type boolean;
            default "false";
            description
                "Indicates whether BFD is enabled on this
                 interface.";
        }
        description
            "BFD client configuration.";
        reference
            "I-D.ietf-bfd-rfc9127-bis: YANG Data Model for
             Bidirectional Forward Detection (BFD).";
    }
    description
        "List of interfaces within the routing instance.";
}
description
    "Interface specific parameters.";
}
uses rib;
}
}
}
```

<CODE ENDS>

```
<CODE BEGINS> file "ietf-bgp-common@2022-03-06.yang"
submodule ietf-bgp-common {
    yang-version 1.1;
    belongs-to ietf-bgp {
        prefix bgp;
    }

    import ietf-bgp-types {
        prefix bt;
        reference
            "RFC XXXX: BGP Model for Service Provider Network.";
    }
    import ietf-inet-types {
        prefix inet;
        reference
            "RFC 6991: Common YANG Data Types.";
    }
    import ietf-bfd-types {
        prefix bfd-types;
        reference
            "RFC XXXX, YANG Data Model for Bidirectional Forward
             Detection.";
    }
}
```

```
}

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

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

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

  Copyright (c) 2021 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

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

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

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

revision 2022-03-06 {
  description
    "Initial Version";
  reference
    "RFC XXXX, BGP Model for Service Provider Network.";
}

grouping neighbor-group-timers-config {
  description
```

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

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

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

      If the configured value of hold-time object was
      a value of (0), then when read this object MUST have a
      value of (0) also.";
    reference
      "RFC 4271, Section 4.2.
      RFC 4271, Section 10.";
  }
  leaf keepalive {
    type uint16 {
      range "0..21845";
    }
    units "seconds";
```

```
description
  "When used as a configuration (rw) value, this Time interval
  (in seconds) for the KeepAlive timer configured for this BGP
  speaker with this peer. A reasonable maximum value for this
  timer would be one-third of the configured hold-time.

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

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

  The actual time interval for the KEEPALIVE messages is
  indicated by operational value of keepalive."
reference
  "RFC 4271, Section 4.4.
  RFC 4271, Section 10.";
}
leaf min-as-origination-interval {
  type uint16 {
    range "0..max";
  }
  units "seconds";
  description
    "Time interval (in seconds) for the MinASOriginationInterval
    timer. The suggested value for this timer is 15 seconds.";
  reference
    "RFC 4271, Section 9.2.1.2.
    RFC 4271, Section 10.";
}
leaf min-route-advertisement-interval {
  type uint16 {
    range "0..max";
  }
  units "seconds";
  description
    "Time interval (in seconds) for the
    MinRouteAdvertisementInterval timer.
    The suggested value for this timer is 30
    seconds for EBGp connections and 5
    seconds for IBGP connections.";
  reference
    "RFC 4271, Section 9.2.1.1.
    RFC 4271, Section 10.";
}
}
```



```
grouping neighbor-group-config {
  description
    "Neighbor level configuration items.";
  leaf peer-as {
    type inet:as-number;
    description
      "AS number of the peer.";
  }
  leaf local-as {
    type inet:as-number;
    description
      "The local autonomous system number that is to be used when
      establishing sessions with the remote peer or peer group, if
      this differs from the global BGP router autonomous system
      number.";
  }

  leaf remove-private-as {
    type bt:remove-private-as-option;
    description
      "When this leaf is specified, remove private AS numbers from
      updates sent to peers.";
  }
  container route-flap-damping {
    if-feature "bt:damping";
    leaf enable {
      type boolean;
      default "false";
      description
        "Enable route flap damping.";
    }
    leaf suppress-above {
      type decimal64 {
        fraction-digits 1;
      }
      default "3.0";
      description
        "This is the value of the instability metric at which
        route suppression takes place. A route is not installed
        in the forwarding information base (FIB), or announced
        even if it is reachable during the period that it is
        suppressed.";
    }
    leaf reuse-above {
      type decimal64 {
        fraction-digits 1;
      }
      default "2.0";
    }
  }
}
```

```
description
    "This is the value of the instability metric at which a
    suppressed route becomes unsuppressed if it is reachable
    but currently suppressed. The value assigned to
    reuse-below must be less than suppress-above.";
}
leaf max-flap {
    type decimal64 {
        fraction-digits 1;
    }
    default "16.0";
    description
        "This is the upper limit of the instability metric. This
        value must be greater than the larger of 1 and
        suppress-above.";
}
leaf reach-decay {
    type uint32;
    units "seconds";
    default "300";
    description
        "This value specifies the time desired for the instability
        metric value to reach one-half of its current value when
        the route is reachable. This half-life value determines
        the rate at which the metric value is decayed. A smaller
        half-life value makes a suppressed route reusable sooner
        than a larger value.";
}
leaf unreach-decay {
    type uint32;
    units "seconds";
    default "900";
    description
        "This value acts the same as reach-decay except that it
        specifies the rate at which the instability metric is
        decayed when a route is unreachable. It should have a
        value greater than or equal to reach-decay.";
}
leaf keep-history {
    type uint32;
    units "seconds";
    default "1800";
    description
        "This value specifies the period over which the route
        flapping history is to be maintained for a given route.
        The size of the configuration arrays described below is
        directly affected by this value.";
}
```

```
        description
            "Routes learned via BGP are subject to weighted route
            dampening.";
    }
    leaf-list send-community {
        if-feature "bt:send-communities";
        type identityref {
            base "bt:send-community-feature";
        }
        description
            "When supported, this tells the router to propagate any
            prefixes that are attached to these community-types.";
    }
    leaf description {
        type string;
        description
            "An optional textual description (intended primarily for use
            with a peer or group";
    }
}

grouping neighbor-group-transport-config {
    description
        "Configuration parameters relating to the transport protocol
        used by the BGP session to the peer.";
    leaf tcp-mss {
        type uint16;
        description
            "Sets the max segment size for BGP TCP sessions.";
    }
    leaf mtu-discovery {
        type boolean;
        default "true";
        description
            "Turns path mtu discovery for BGP TCP sessions on (true) or
            off (false).";
        reference
            "RFC 1191: Path MTU discovery.";
    }
    leaf passive-mode {
        type boolean;
        default "false";
        description
            "Wait for peers to issue requests to open a BGP session,
            rather than initiating sessions from the local router.";
    }
    leaf local-address {
        type union {
```

```
    type inet:ip-address;
    type leafref {
      path "../../../../../interfaces/interface/name";
    }
  }
  description
    "Set the local IP (either IPv4 or IPv6) address to use for
    the session when sending BGP update messages. This may be
    expressed as either an IP address or reference to the name
    of an interface.";
}
leaf md5-auth-password {
  type string;
  description
    "Configures an MD5 authentication password for use with
    neighboring devices.";
  reference
    "RFC 2385: Protection of BGP Sessions via the TCP MD5
    Signature Option.";
}
container bfd {
  if-feature "bt:bfd";
  uses bfd-types:client-cfg-parms;
  description
    "BFD client configuration.";
  reference
    "RFC XXXX, YANG Data Model for Bidirectional Forwarding
    Detection.";
}
}

grouping graceful-restart-config {
  description
    "Configuration parameters relating to BGP graceful restart.";
  leaf enabled {
    type boolean;
    default "false";
    description
      "Enable or disable the graceful-restart capability.";
  }
  leaf restart-time {
    type uint16 {
      range "0..4096";
    }
    description
      "Estimated time (in seconds) for the local BGP speaker to
      restart a session. This value is advertise in the graceful
      restart BGP capability. This is a 12-bit value, referred to
```

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

grouping global-group-use-multiple-paths {
    description
        "Common grouping used for both global and groups which provides
        configuration and state parameters relating to use of multiple
        paths";
    container use-multiple-paths {
        description
            "Parameters related to the use of multiple paths for the
            same NLRI";
        leaf enabled {
            type boolean;
            default "false";
            description
                "Whether the use of multiple paths for the same NLRI is
                enabled for the neighbor. This value is overridden by any
                more specific configuration value.";
        }
    }
}
```

```
    container ebgp {
      description
        "Multi-Path parameters for eBGP";
      leaf allow-multiple-as {
        type boolean;
        default "false";
        description
          "Allow multi-path to use paths from different neighboring
          ASes. The default is to only consider multiple paths
          from the same neighboring AS.";
      }
      leaf maximum-paths {
        type uint32;
        default "1";
        description
          "Maximum number of parallel paths to consider when using
          BGP multi-path. The default is use a single path.";
      }
    }
  }
  container ibgp {
    description
      "Multi-Path parameters for iBGP";
    leaf maximum-paths {
      type uint32;
      default "1";
      description
        "Maximum number of parallel paths to consider when using
        iBGP multi-path. The default is to use a single path";
    }
  }
}

grouping route-selection-options {
  description
    "Configuration and state relating to route selection options";
  container route-selection-options {
    description
      "Parameters relating to options for route selection";
    leaf always-compare-med {
      type boolean;
      default "false";
      description
        "Compare multi-exit discriminator (MED) value from
        different ASes when selecting the best route. The default
        behavior is to only compare MEDs for paths received from
        the same AS.";
    }
  }
}
```

```
leaf ignore-as-path-length {
  type boolean;
  default "false";
  description
    "Ignore the AS path length when selecting the best path.
     The default is to use the AS path length and prefer paths
     with a shorter length.";
}
leaf external-compare-router-id {
  type boolean;
  default "true";
  description
    "When comparing similar routes received from external BGP
     peers, use the router-id as a criterion to select the
     active path.";
}
leaf advertise-inactive-routes {
  type boolean;
  default "false";
  description
    "Advertise inactive routes to external peers. The default
     is to only advertise active routes.";
  reference
    "I-D.ietf-idr-best-external: Advertisement of the best
     external route in BGP.";
}
leaf enable-aigp {
  type boolean;
  default "false";
  description
    "Flag to enable sending / receiving accumulated IGP
     attribute in routing updates";
  reference
    "RFC 7311: AIGP Metric Attribute for BGP.";
}
leaf ignore-next-hop-igp-metric {
  type boolean;
  default "false";
  description
    "Ignore the IGP metric to the next-hop when calculating BGP
     best-path. The default is to select the route for which
     the metric to the next-hop is lowest";
}
leaf enable-med {
  type boolean;
  default "false";
  description
    "Flag to enable sending/receiving of MED metric attribute
```

```
        in routing updates.";
    }
    container med-plus-igp {
        leaf enabled {
            type boolean;
            default "false";
            description
                "When enabled allows BGP to use MED and IGP values
                 defined below to determine the optimal route.";
            reference
                "RFC 4451: BGP MED Considerations.";
        }
        leaf igp-multiplier {
            type uint16;
            default 1;
            description
                "Specifies an IGP cost multiplier.";
            reference
                "RFC 4451: BGP MED Considerations.";
        }
        leaf med-multiplier {
            type uint16;
            default 1;
            description
                "Specifies a MED multiplier.";
            reference
                "RFC 4451: BGP MED Considerations.";
        }
        description
            "The med-plus-igp option enables BGP to use the sum of
             MED multiplied by a MED multiplier and IGP cost multiplied
             by IGP cost multiplier to select routes when MED is
             required to determine the optimal route.";
    }
}

grouping state {
    description
        "Grouping containing common counters relating to prefixes and
         paths";
    leaf total-paths {
        type uint32;
        config false;
        description
            "Total number of BGP paths (BGP routes) within the context";
    }
    leaf total-prefixes {
```



```
        type uint32;
        config false;
        description
            "Total number of BGP prefixes (destinations) received within
            the context";
    }
}
}
<CODE ENDS>

<CODE BEGINS> file "ietf-bgp-common-multiprotocol@2022-03-06.yang"
submodule ietf-bgp-common-multiprotocol {
    yang-version 1.1;
    belongs-to ietf-bgp {
        prefix bgp;
    }

    import ietf-bgp-types {
        prefix bt;
    }
    import ietf-routing-policy {
        prefix rt-pol;
    }
    import ietf-routing-types {
        prefix rt-types;
    }
    include ietf-bgp-common;

    // meta

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

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

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

        Copyright (c) 2021 IETF Trust and the persons identified as
        authors of the code. All rights reserved."
```

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

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

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

```
revision 2022-03-06 {
  description
    "Initial Version";
  reference
    "RFC XXX, BGP Model for Service Provider Network.";
}

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

grouping mp-afi-safi-config {
  description
    "Configuration parameters used for all BGP AFI-SAFIs";
  leaf name {
    type identityref {
      base bt:afi-safi-type;
    }
    description
      "AFI, SAFI";
  }
}
```

```
    leaf enabled {
      type boolean;
      default "false";
      description
        "This leaf indicates whether this AFI,SAFI is enabled for
        the neighbor or group";
    }
  }

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

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

```
        "Multicast IPv4 L3VPN configuration options";
    // include common L3VPN multicast options
    uses mp-l3vpn-ipv4-ipv6-multicast-common;
    // placeholder for IPv4 Multicast L3VPN specific configuration
    // options
}
container l3vpn-ipv6-multicast {
    when "../name = 'bt:l3vpn-ipv6-multicast'" {
        description
            "Include this container for multicast IPv6 L3VPN specific
            configuration";
    }
    description
        "Multicast IPv6 L3VPN configuration options";
    // include common L3VPN multicast options
    uses mp-l3vpn-ipv4-ipv6-multicast-common;
    // placeholder for IPv6 Multicast L3VPN specific configuration
    // options
}
container l2vpn-vpls {
    when "../name = 'bt:l2vpn-vpls'" {
        description
            "Include this container for BGP-signalled VPLS specific
            configuration";
    }
    description
        "BGP-signalled VPLS configuration options";
    // include common L2VPN options
    uses mp-l2vpn-common;
    // placeholder for BGP-signalled VPLS specific configuration
    // options
}
container l2vpn-evpn {
    when "../name = 'bt:l2vpn-evpn'" {
        description
            "Include this container for BGP EVPN specific
            configuration";
    }
    description
        "BGP EVPN configuration options";
    // include common L2VPN options
    uses mp-l2vpn-common;
    // placeholder for BGP EVPN specific configuration options
}
}

// Common groupings across multiple AFI,SAFIs
```

```
grouping mp-all-afi-safi-common {
  description
    "Grouping for configuration common to all AFI,SAFI";
  container prefix-limit {
    description
      "Parameters relating to the prefix limit for the AFI-SAFI";
    leaf max-prefixes {
      type uint32;
      description
        "Maximum number of prefixes that will be accepted from the
        neighbor";
    }
    leaf shutdown-threshold-pct {
      type rt-types:percentage;
      description
        "Threshold on number of prefixes that can be received from
        a neighbor before generation of warning messages or log
        entries. Expressed as a percentage of max-prefixes";
    }
    leaf restart-timer {
      type uint32;
      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 mp-ipv4-ipv6-unicast-common {
  description
    "Common configuration that is applicable for IPv4 and IPv6
    unicast";
  // include common afi-safi options.
  uses mp-all-afi-safi-common;
  // configuration options that are specific to IPv[46] unicast
  leaf send-default-route {
    type boolean;
    default "false";
    description
      "If set to true, send the default-route to the neighbor(s)";
  }
}

grouping mp-l3vpn-ipv4-ipv6-unicast-common {
  description
    "Common configuration applied across L3VPN for IPv4
```

```
        and IPv6";
        // placeholder -- specific configuration options that are generic
        // across IPv[46] unicast address families.
        uses mp-all-afi-safi-common;
    }

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

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

    // Config groupings for common groups

    grouping mp-all-afi-safi-common-prefix-limit-config {
        description
            "Configuration parameters relating to prefix-limits for an
            AFI-SAFI";
    }
}
<CODE ENDS>

<CODE BEGINS> file "ietf-bgp-common-structure@2022-03-06.yang"
submodule ietf-bgp-common-structure {
    yang-version 1.1;
    belongs-to ietf-bgp {
        prefix bgp;
    }

    import ietf-routing-policy {
        prefix rt-pol;
        reference
            "RFC ZZZZ, A YANG Data Model for Routing Policy Management";
    }
    import ietf-bgp-types {
        prefix bt;
    }
}
```

```
reference
  "RFC XXXX, BGP YANG Model for Service Provider Network.";
}
include ietf-bgp-common-multiprotocol;
include ietf-bgp-common;

// meta

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

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

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

  Copyright (c) 2021 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

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

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

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

revision 2022-03-06 {
  description
    "Initial Version";
  reference
```



```
    "RFC XXX, BGP Model for Service Provider Network.";
  }

  grouping structure-neighbor-group-logging-options {
    description
      "Structural grouping used to include error handling
       configuration and state for both BGP neighbors and groups";
    container logging-options {
      description
        "Logging options for events related to the BGP neighbor or
         group";
      leaf log-neighbor-state-changes {
        type boolean;
        default "true";
        description
          "Configure logging of peer state changes. Default is to
           enable logging of peer state changes.

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

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

```
    }  
  }  
  
  grouping structure-neighbor-group-route-reflector {  
    description  
      "Structural grouping used to include route reflector  
      configuration and state for both BGP neighbors and peer  
      groups";  
    container route-reflector {  
      description  
        "Route reflector parameters for the BGP peer-group";  
      reference  
        "RFC 4456: BGP Route Reflection.";   
      leaf cluster-id {  
        type bt:rr-cluster-id-type;  
        description  
          "Route Reflector cluster id to use when local router is  
          configured as a route reflector. Commonly set at the  
          group level, but allows a different cluster id to be set  
          for each neighbor.";   
        reference  
          "RFC 4456: BGP Route Reflection: An Alternative to  
          Full Mesh.";   
      }  
      leaf no-client-reflect {  
        type boolean;  
        default "false";  
        description  
          "When set to 'true', this disables route redistribution  
          by the Route Reflector. It is set 'true' when the client  
          is fully meshed in its peer-group to prevent sending of  
          redundant route advertisements.";   
      }  
      leaf client {  
        type boolean;  
        default "false";  
        description  
          "Configure the neighbor as a route reflector client.";   
        reference  
          "RFC 4456: BGP Route Reflection: An Alternative to  
          Full Mesh.";   
      }  
    }  
  }  
  
  grouping structure-neighbor-group-as-path-options {  
    description  
      "Structural grouping used to include AS_PATH manipulation
```

```
    configuration and state for both BGP neighbors and peer
    groups";
  container as-path-options {
    description
      "AS_PATH manipulation parameters for the BGP neighbor or
      group";
    leaf allow-own-as {
      type uint8;
      default "0";
      description
        "Specify the number of occurrences of the local BGP
        speaker's AS that can occur within the AS_PATH before it
        is rejected as looped.";
    }
    leaf replace-peer-as {
      type boolean;
      default "false";
      description
        "Replace occurrences of the peer's AS in the AS_PATH with
        the local autonomous system number";
    }
  }
}

grouping structure-neighbor-group-add-paths {
  description
    "Structural grouping used to include ADD-PATHs configuration
    and state for both BGP neighbors and peer groups";
  container add-paths {
    if-feature "bt:add-paths";
    description
      "Parameters relating to the advertisement and receipt of
      multiple paths for a single NLRI (add-paths)";
    reference
      "RFC 7911: Advertisements of Multiple Paths in BGP.";
    leaf receive {
      type boolean;
      default "false";
      description
        "Enable ability to receive multiple path advertisements for
        an NLRI from the neighbor or group";
    }
    choice send {
      description
        "Choice of sending the max. number of paths or to send
        all.";
      case max {
        leaf max {
```

```

        type uint8;
        description
            "The maximum number of paths to advertise to neighbors
             for a single NLRI";
    }
}
case all {
    leaf all {
        type empty;
        description
            "Send all the path advertisements to neighbors for a
             single NLRI.";
    }
}
leaf eligible-prefix-policy {
    type leafref {
        path "/rt-pol:routing-policy/rt-pol:policy-definitions/"
          + "rt-pol:policy-definition/rt-pol:name";
    }
    description
        "A reference to a routing policy which can be used to
         restrict the prefixes for which add-paths is enabled";
}
}
}
}
<CODE ENDS>

<CODE BEGINS> file "ietf-bgp-neighbor@2022-03-06.yang"
submodule ietf-bgp-neighbor {
    yang-version 1.1;
    belongs-to ietf-bgp {
        prefix bgp;
    }

    import ietf-bgp-types {
        prefix bt;
        reference
            "RFC XXXX, BGP Model for Service Provider Network.";
    }

    // Include the common submodule

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

```

```
// meta

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

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

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

  Copyright (c) 2021 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

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

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

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

revision 2022-03-06 {
  description
    "Initial Version";
  reference
    "RFC XXX, BGP Model for Service Provider Network.";
}

grouping bgp-neighbor-use-multiple-paths {
  description
    "Multi-path configuration and state applicable to a BGP
    neighbor";
```

```
    container use-multiple-paths {
      description
        "Parameters related to the use of multiple-paths for the same
        NLRI when they are received only from this neighbor";
      leaf enabled {
        type boolean;
        default "false";
        description
          "Whether the use of multiple paths for the same NLRI is
          enabled for the neighbor.";
      }
      container ebgp {
        description
          "Multi-path configuration for eBGP";
        leaf allow-multiple-as {
          type boolean;
          default "false";
          description
            "Allow multi-path to use paths from different neighboring
            ASes. The default is to only consider multiple paths
            from the same neighboring AS.";
        }
      }
    }
  }
}

grouping bgp-neighbor-counters-message-types-state {
  description
    "Grouping of BGP message types, included for re-use across
    counters";
  leaf updates-received {
    type uint64;
    description
      "Number of BGP UPDATE messages received from this neighbor.";
    reference
      "RFC 4273: bgpPeerInUpdates.";
  }
  leaf updates-sent {
    type uint64;
    description
      "Number of BGP UPDATE messages sent to this neighbor";
    reference
      "RFC 4273 - bgpPeerOutUpdates";
  }
  leaf messages-received {
    type uint64;
    description
      "Number of BGP messages received from thsi neighbor";
  }
}
```

```
        reference
          "RFC 4273 - bgpPeerInTotalMessages";
      }
      leaf messages-sent {
        type uint64;
        description
          "Number of BGP messages received from this neighbor";
        reference
          "RFC 4273 - bgpPeerOutTotalMessages";
      }
      leaf notification {
        type uint64;
        description
          "Number of BGP NOTIFICATION messages indicating an error
           condition has occurred exchanged.";
      }
    }
  }

  grouping bgp-neighbor-afi-safi-list {
    description
      "List of address-families associated with the BGP neighbor";
    list afi-safi {
      key "name";
      description
        "AFI, SAFI configuration available for the neighbor or
         group";
      uses mp-afi-safi-config;
      leaf active {
        type boolean;
        config false;
        description
          "This value indicates whether a particular AFI-SAFI has
           been successfully negotiated with the peer. An AFI-SAFI
           may be enabled in the current running configuration, but
           a session restart may be required in order to negotiate
           the new capability.";
      }
      container prefixes {
        config false;
        description
          "Prefix counters for the AFI/SAFI in this BGP session";
        leaf received {
          type 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";
    }
}
container graceful-restart {
    if-feature "bt:graceful-restart";
    description
        "Parameters relating to BGP graceful-restart";
    uses mp-afi-safi-graceful-restart-config;
    leaf received {
        type boolean;
        config false;
        description
            "This leaf indicates whether the neighbor advertised the
            ability to support graceful-restart for this AFI-SAFI";
    }
    leaf advertised {
        type boolean;
        config false;
        description
            "This leaf indicates whether the ability to support
            graceful-restart has been advertised to the peer";
    }
    leaf local-forwarding-state-preserved {
        type boolean;
        config false;
        description
            "This leaf indicates whether the local router has
            or would advertise the Forwarding State bit in its
            Graceful Restart capability for this AFI-SAFI.";
        reference
            "RFC 4724: Graceful Restart Mechanism for BGP.";
    }
    leaf forwarding-state-preserved {
        type boolean;
        config false;
        description
            "This leaf indicates whether the neighbor has advertised
            the Forwarding State bit in its Graceful Restart
            capability for this AFI-SAFI.";
        reference
            "RFC 4724: Graceful Restart Mechanism for BGP.";
```



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

7.2. BGP types

```
<CODE BEGINS> file "ietf-bgp-types@2022-03-06.yang"
module ietf-bgp-types {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-bgp-types";
  prefix bt;

  import ietf-inet-types {
    prefix inet;
  }

  // meta

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

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

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

    Copyright (c) 2021 IETF Trust and the persons identified as
```

authors of the code. All rights reserved.

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

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

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

```
revision 2022-03-06 {
  description
    "Initial Version";
  reference
    "RFC XXX, BGP Model for Service Provider Network.";
}

/*
 * Features.
 */

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

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

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

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

```
}

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

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

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

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

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

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

/*
 * Identities.
```

```
*/

identity bgp-capability {
  description
    "Base identity for a BGP capability";
}

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

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

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

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

identity add-paths {
  if-feature "add-paths";
  base bgp-capability;
  description
    "Advertisement of multiple paths for the same address prefix
      without the new paths implicitly replacing any previous
      ones.";
```

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

  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: Multiprotocol Extensions for BGP-4";
  }

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

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

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

  identity l3vpn-ipv4-unicast {
    base afi-safi-type;
    description
      "Unicast IPv4 MPLS L3VPN (AFI,SAFI = 1,128)";
    reference
```

```
    "RFC 4364: BGP/MPLS IP Virtual Private Networks (VPNs).";
  }

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

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

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

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

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

  identity bgp-well-known-std-community {
    description
      "Base identity for reserved communities within the standard
```

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

identity no-export {
    base bgp-well-known-std-community;
    description
        "Do not export NLRI received carrying this community outside
        the bounds of this autonomous system, or this confederation
        (if the local autonomous system is a confederation member AS).
        This community has a value of 0xFFFF0001.";
    reference
        "RFC 1997: BGP Communities Attribute.";
}

identity no-advertise {
    base bgp-well-known-std-community;
    description
        "All NLRI received carrying this community must not be
        advertised to other BGP peers. This community has a value of
        0xFFFF0002.";
    reference
        "RFC 1997: BGP Communities Attribute.";
}

identity no-export-subconfed {
    base bgp-well-known-std-community;
    description
        "All NLRI received carrying this community must not be
        advertised to external BGP peers - including over
        confederation sub-AS boundaries. This community has a value of
        0xFFFF0003.";
    reference
        "RFC 1997: BGP Communities Attribute.";
}

identity no-peer {
    base bgp-well-known-std-community;
    description
        "An autonomous system receiving NLRI tagged with this community
        is advised not to re-advertise the NLRI to external bilateral
        peer autonomous systems. An AS may also filter received NLRI
        from bilateral peer sessions when they are tagged with this
        community value. This community has a value of 0xFFFF0004.";
    reference
        "RFC 3765: NOPEER Community for BGP.";
```

```
}

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

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

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

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

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

identity send-community-feature {
  description
```



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

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

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

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

/*
 * Typedefs.
 */

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

typedef bgp-well-known-community-type {
    type identityref {
```

```

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

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

typedef bgp-ext-community-type {
  type union {
    type string {
      // Type 1: 2-octet global and 4-octet local
      //           (AS number)           (Integer)
      pattern '(6[0-5][0-5][0-3][0-5]|[1-5][0-9]{4})|'
        + '[1-9][0-9]{1,4}|[0-9]):'
        + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
        + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[1-9])';
    }

    type string {
      // Type 2: 4-octet global and 2-octet local
      //           (ipv4-address)       (integer)
      pattern '((([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|'
        + '25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|'
        + '2[0-4][0-9]|25[0-5]):'
        + '(6[0-5][0-5][0-3][0-5]|[1-5][0-9]{4})|'
        + '[1-9][0-9]{1,4}|[0-9])';
    }
  }
}

```

```

type string {
    // route-target with Type 1
    // route-target:(ASN):(local-part)
    // 2 octets global and 4 octets local.
    pattern 'route\-target:(6[0-5][0-5][0-3][0-5]|'
        + '[1-5][0-9]{4}|[1-9][0-9]{1,4}|[0-9]):'
        + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
        + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[1-9])';
}

type string {
    // route-target with Type 2
    // route-target:(IPv4):(local-part)
    // 4 bytes of IP address, and 2 bytes for local.
    pattern 'route\-target:'
        + '(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|'
        + '25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|'
        + '2[0-4][0-9]|25[0-5]):'
        + '(6[0-5][0-5][0-3][0-5]|[1-5][0-9]{4}|'
        + '[1-9][0-9]{1,4}|[0-9])';
}

type string {
    // route-origin with Type 1
    // All 6 octets are open.
    pattern 'route\-origin:(6[0-5][0-5][0-3][0-5]|'
        + '[1-5][0-9]{4}|[1-9][0-9]{1,4}|[0-9]):'
        + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
        + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[1-9])';
}

type string {
    // route-origin with Type 2
    // 4 octets of IP address and two octets of local.
    pattern 'route\-origin:'
        + '(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|'
        + '25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|'
        + '2[0-4][0-9]|25[0-5]):'
        + '(6[0-5][0-5][0-3][0-5]|[1-5][0-9]{4}|'
        + '[1-9][0-9]{1,4}|[0-9])';
}
}
description
    "Type definition for extended community attributes";
reference
    "RFC 4360 - BGP Extended Communities Attribute";
}

```

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

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

typedef bgp-large-community-type {
  type string {
    // 4-octets global:4-octets local part-1:4-octets local part-2.
    pattern '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
      + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[1-9]):'
      + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
      + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[1-9]):'
      + '(4[0-2][0-9][0-4][0-9][0-6][0-7][0-2][0-9][0-6]|'
      + '[1-3][0-9]{9}|[1-9]([0-9]{1,7})?[0-9]|[1-9])'
      + '';
  }
  description
    "Type definition for a large BGP community";
  reference
    "RFC 8092: BGP Large Communities Attribute.";
}

typedef peer-type {
  type enumeration {
    enum internal {
      description
```

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

identity remove-private-as-option {
    description
        "Base identity for options for removing private autonomous
        system numbers from the AS_PATH attribute";
}

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

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

```
typedef remove-private-as-option {
  type identityref {
    base remove-private-as-option;
  }
  description
    "Set of options for configuring how private AS path numbers
    are removed from advertisements";
}

typedef 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";
}
}
<CODE ENDS>
```

7.3. BGP policy data

```
<CODE BEGINS> file "ietf-bgp-policy@2022-03-06.yang"
module ietf-bgp-policy {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-bgp-policy";
  prefix bp;

  // import some basic types

  import ietf-inet-types {
    prefix inet;
  }
  import ietf-routing-policy {
    prefix rt-pol;
  }
  import ietf-bgp-types {
    prefix bt;
  }
  import ietf-routing-types {
    prefix rt-types;
  }

  organization
    "IETF IDR Working Group";
  contact
```

"WG Web: <<http://datatracker.ietf.org/wg/idr>>
WG List: <idr@ietf.org>

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

description

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

Copyright (c) 2022 IETF Trust and the persons identified as
authors of the code. All rights reserved.

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

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

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

```
revision 2022-03-06 {  
  description  
    "Initial Version";  
  reference  
    "RFC XXX, BGP Model for Service Provider Network.";  
}
```

```
/*  
 * typedef statements  
 */
```

```
typedef bgp-set-community-option-type {  
  type enumeration {  
    enum add {  
      description
```

```

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

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

typedef bgp-set-med-type {
    type union {
        type uint32;
        type string {
            pattern '^[+-]([0-9]{1,8}|[0-3][0-9]{1,9}|4[0-1][0-9]{1,8}|
                + '428[0-9]{1,7}|429[0-3][0-9]{1,6}|42948[0-9]{1,5}|
                + '42949[0-5][0-9]{1,4}|429496[0-6][0-9]{1,3}|
                + '4294971[0-9]{1,2}|42949728[0-9]|42949729[0-5]))$';
        }
        type enumeration {
            enum igp {
                description

```



```
        "Set the MED value to the IGP cost toward the
        next hop for the route.";
    }
    enum med-plus-igp {
        description
            "Before comparing MED values for path selection, adds to
            the MED the cost of the IGP route to the BGP next-hop
            destination.

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

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

// Identities

// augment statements

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

```
    leaf name {
      type string;
      description
        "Name / label of the community set -- this is used to
         reference the set in match conditions.";
    }
    leaf-list member {
      type union {
        type bt:bgp-std-community-type;
        type bt:bgp-community-regexp-type;
        type bt:bgp-well-known-community-type;
      }
      description
        "Members of the community set";
    }
  }
}

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

container large-community-sets {
  description
    "Enclosing container for list of large BGP community
     sets";
  list large-community-set {
```

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

container as-path-sets {
  description
    "Enclosing container for list of define AS path sets.";
  list as-path-set {
    key "name";
    description
      "List of defined AS path sets.";
    leaf name {
      type string;
      description
        "Name of the AS path set -- this is used to reference
        the set in match conditions.";
    }
    leaf-list member {
      type string;
      description
        "AS path regular expression -- list of ASes in the
        set.";
    }
  }
}

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

  list next-hop-set {
```

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

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

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

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

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

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

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

```
}

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

leaf local-pref-eq {
  type uint32;
  description
    "Condition to check if the local pref attribute is equal to
    the specified value.";
}

leaf-list neighbor-eq {
  type inet:ip-address;
  description
    "List of neighbor addresses to check for in the ingress
    direction.";
}

leaf route-type {
  type enumeration {
    enum internal {
      description
        "route type is internal.";
    }
    enum external {
      description
        "route type is external.";
    }
  }
  description
    "Condition to check the route type in the route update.";
}

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

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

```
        update.";
    }

    choice operation {
        case eq {
            leaf eq {
                type empty;
                description
                    "Check to see if the value is equal.";
            }
        }

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

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

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

        The as-path-length SHALL be calculated and SHALL follow
        RFC 4271 rules.";
    reference
        "RFC 4271: BGP-4.";

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

```
choice operation {
  case eq {
    leaf eq {
      type empty;
      description
        "Check to see if the value is equal.";
    }
  }

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

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

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

container match-ext-community-set {
```

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

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

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

container match-next-hop-set {
```



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

augment "/rt-pol:routing-policy/rt-pol:policy-definitions/" +
  "rt-pol:policy-definition/rt-pol:statements/" +
  "rt-pol:statement/rt-pol:actions" {
  description
    "BGP policy actions added to routing policy module.";
  container bgp-actions {
    description
      "Top-level container for BGP-specific actions";
    leaf set-route-origin {
      type bt:bgp-origin-attr-type;
      description
        "Set the origin attribute to the specified value";
    }
    leaf set-local-pref {
      type uint32;
      description
        "Set the local pref attribute on the route.";
    }
    leaf set-next-hop {
      type bgp-next-hop-type;
      description
        "Set the next-hop attribute in the route.";
    }
    leaf set-med {
      type bgp-set-med-type;
      description
        "Set the med metric attribute in the route.";
    }
    container set-as-path-prepend {
      description
        "Action to prepend local AS number to the AS-path a
```

```
        specified number of times";

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

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

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

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

        case reference {
            leaf community-set-ref {
                type leafref {
                    path "/rt-pol:routing-policy/rt-pol:defined-sets/"
                }
            }
        }
    }
}
```

```
        + "bgp-defined-sets/"
        + "community-sets/community-set/name";
    }
    description
        "References a defined community set by name";
    }
}

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

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

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

```

    }
  }
}

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

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

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

```

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

7.4. RIB modules

```
<CODE BEGINS> file "ietf-bgp-rib@2022-03-06.yang"  
submodule ietf-bgp-rib {  
  yang-version 1.1;  
  belongs-to ietf-bgp {  
    prefix br;  
  }  
  
  /*  
   * Import and Include  
   */  
  
  import ietf-bgp-types {  
    prefix bt;  
    reference  
      "RFC XXXX: BGP YANG Model for Service Provider Networks.";  
  }  
  import ietf-inet-types {  
    prefix inet;  
    reference  
      "RFC 6991: Common YANG Types.";  
  }  
  import ietf-yang-types {  
    prefix yang;  
    reference  
      "RFC 6991: Common YANG Types.";  
  }  
  import ietf-routing-types {  
    prefix rt;  
    reference  
      "RFC 8294: Routing Area YANG Types.";  
  }  
  include ietf-bgp-rib-types;  
  include ietf-bgp-rib-tables;  
  
  // groupings of attributes in three categories:  
  // - shared across multiple routes  
  // - common to LOC-RIB and Adj-RIB, but not shared across routes  
  // - specific to LOC-RIB or Adj-RIB  
  // groupings of annotations for each route or table  
  include ietf-bgp-rib-attributes;  
  
  organization
```

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

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

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

  loc-rib: This is the main BGP routing table for the local
  routing instance, containing best-path selections for each
  prefix. The loc-rib table may contain multiple routes for a
  given prefix, with an attribute to indicate which was selected
  as the best path. Note that multiple paths may be used or
  advertised even if only one path is marked as best, e.g., when
  using BGP add-paths. An implementation may choose to mark
  multiple paths in the RIB as best path by setting the flag to
  true for multiple entries.

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

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

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

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

  Copyright (c) 2021 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject to
```

the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>).

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

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

```
revision 2022-03-06 {
  description
    "Initial Version";
  reference
    "RFC XXXX, BGP YANG Model for Service Provider Network.";
}
```

```
grouping attr-set-attributes {
  description
    "A grouping for all attribute set parameters.";

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

    leaf origin {
      type bt:bgp-origin-attr-type;
      description
        "BGP attribute defining the origin of the path
        information.";
    }
    leaf atomic-aggregate {
      type boolean;
      description
        "BGP attribute indicating that the prefix is an atomic
        aggregate; i.e., the peer selected is a less specific
        route without selecting a more specific route that is
        subsumed by it.";
      reference
        "RFC 4271: Section 5.1.6.";
    }
  }
  leaf next-hop {
    type inet:ip-address;
```

```
    description
      "BGP next hop attribute defining the IP address of the
       router that should be used as the next hop to the
       destination.";
    reference
      "RFC 4271: Section 5.1.3.";
  }
  leaf link-local-next-hop {
    type inet:ipv6-address;
    description
      "When both a global and a link-local next-hop are sent
       when following RFC 2545 procedures, this leaf contains
       the link-local next-hop.";
    reference
      "RFC 2545: Use of BGP-4 Multiprotocol Extensions for IPv6
       Inter-Domain Routing";
  }
  leaf med {
    type uint32;
    description
      "BGP multi-exit discriminator attribute used in the BGP
       route selection process.";
    reference
      "RFC 4271: Section 5.1.4.";
  }
  leaf local-pref {
    type uint32;
    description
      "BGP local preference attribute sent to internal peers to
       indicate the degree of preference for externally learned
       routes. The route with the highest local preference
       value is preferred.";
    reference
      "RFC 4271: Section 5.1.5.";
  }
  leaf originator-id {
    type yang:dotted-quad;
    description
      "BGP attribute that provides the id as an IPv4 address
       of the originator of the announcement.";
    reference
      "RFC 4456 - BGP Route Reflection: An Alternative to Full
       Mesh Internal BGP (IBGP)";
  }
  leaf-list cluster-list {
    type yang:dotted-quad;
    description
      "Represents the reflection path that the route has
```



```
        passed.";
    reference
        "RFC 4456 - BGP Route Reflection: An Alternative to Full
        Mesh Internal BGP (IBGP)";
}
leaf aigp-metric {
    type uint64;
    description
        "BGP path attribute representing the accumulated IGP
        metric for the path";
    reference
        "RFC 7311 - The Accumulated IGP Metric Attribute for BGP";
}
container aggregator {
    config false;
    description
        "BGP attribute indicating the prefix has been
        aggregated by the specified AS and router.";
    reference
        "RFC 4271: Section 5.1.7.
        RFC 6793 - BGP Support for Four-octet AS Number Space.";
    leaf as {
        type inet:as-number;
        description
            "AS number of the autonomous system that performed the
            aggregation.";
    }
    leaf address {
        type inet:ipv4-address;
        description
            "IP address of the router that performed the
            aggregation.";
    }
}
container aggregator4 {
    config false;
    description
        "BGP attribute indicating the prefix has been
        aggregated by the specified AS and router.
        This value is populated with the received or sent
        attribute in Adj-RIB-In or Adj-RIB-Out, respectively.
        It should not be populated in Loc-RIB since the Loc-RIB
        is expected to store the effective AGGREGATOR in the
        aggregator/as leaf regardless of being 4-octet or
        2-octet.";
    reference
        "RFC 4271: Section 5.1.7.";
    leaf as4 {
```

```
type inet:as-number;
description
  "AS number of the autonomous system that performed the
  aggregation (4-octet representation). This value is
  populated if an upstream router is not 4-octet capable.
  Its semantics are similar to the AS4_PATH optional
  transitive attribute";
reference
  "RFC 6793 - BGP Support for Four-octet AS Number Space";
}
leaf address {
  type inet:ipv4-address;
  description
    "IP address of the router that performed the
    aggregation.";
}
}
container as-path {
  description
    "Enclosing container for the list of AS path segments.

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

    In the Loc-RIB, this list should reflect the effective
    AS path for the route, e.g., a 4-octet value if the
    local router is 4-octet capable.";
  reference
    "RFC 4271 - A Border Gateway Protocol 4 (BGP-4)
    RFC 6793 - BGP Support for Four-octet AS Number Space
    RFC 5065 - Autonomous System Confederations for BGP";
  list segment {
    config false;
    uses bgp-as-path-attr;
    description
      "List of AS PATH segments";
  }
}
}
container as4-path {
  description
    "This is the path encoded with 4-octet
    AS numbers in the optional transitive AS4_PATH attribute.
    This value is populated with the received or sent
    attribute in Adj-RIB-In or Adj-RIB-Out, respectively.
    It should not be populated in Loc-RIB since the Loc-RIB
```

```
        is expected to store the effective AS-Path in the
        as-path leaf regardless of being 4-octet or 2-octet.";
    reference
        "RFC 6793 - BGP Support for Four-octet AS Number Space";
    list segment {
        config false;
        uses bgp-as-path-attr;
        description
            "List of AS PATH segments";
    }
}

grouping attr-set {
    description
        "A grouping for all path attributes.";

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

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

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

grouping ext-community-attributes {
    description
```

```
    "A grouping for all extended community parameters.";

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

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

    leaf-list large-community {
        type bt:bgp-large-community-type;
        description
            "List of BGP large community attributes.";
        reference
            "RFC 8092: BGP Large Communities Attribute.";
    }
}

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

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

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

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

```
        uses bgp-loc-rib-common-keys;
        uses bgp-loc-rib-common-attr-refs;
        uses bgp-common-route-annotations-state;
        uses bgp-unknown-attr-top;
        uses rib-ext-route-annotations;
    }
}

container neighbors {
    config false;
    description
        "Enclosing container for neighbor list.";
    list neighbor {
        key "neighbor-address";
        description
            "List of neighbors (peers) of the local BGP
            speaker.";
        leaf neighbor-address {
            type inet:ip-address;
            description
                "IP address of the BGP neighbor or peer.";
        }
        container adj-rib-in-pre {
            description
                "Per-neighbor table containing the NLRI updates
                received from the neighbor before any local
                input policy rules or filters have been applied.
                This can be considered the 'raw' updates from
                the neighbor.";
            uses ipv4-adj-rib-common;
            uses clear-routes {
                description
                    "Clears the adj-rib-in state for the containing
                    neighbor. Subsequently, implementations might
                    issue a 'route refresh' if 'route refresh' has
                    been negotiated, or reset the session. ";
            }
        }
        container adj-rib-in-post {
            description
                "Per-neighbor table containing the paths received
                from the neighbor that are eligible for
                best-path selection after local input policy
                rules have been applied.";
            uses ipv4-adj-rib-in-post;
            uses clear-routes {
                description
```

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

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

    container loc-rib {
```



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

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

```

```
    "Per-neighbor table containing the NLRI updates
    received from the neighbor before any local
    input policy rules or filters have been applied.
    This can be considered the 'raw' updates from
    the neighbor.";
  uses ipv6-adj-rib-common;
  uses clear-routes {
    description
      "Clears the adj-rib-in state for the containing
      neighbor. Subsequently, implementations might
      issue a 'route refresh' if 'route refresh' has
      been negotiated, or reset the session. ";
  }
}
container adj-rib-in-post {
  description
    "Per-neighbor table containing the paths received
    from the neighbor that are eligible for
    best-path selection after local input policy
    rules have been applied.";
  uses ipv6-adj-rib-in-post;
  uses clear-routes {
    description
      "Clears the adj-rib-in state for the containing
      neighbor. Subsequently, implementations might
      issue a 'route refresh' if 'route refresh' has
      been negotiated, or reset the session. ";
  }
}
container adj-rib-out-pre {
  description
    "Per-neighbor table containing paths eligible for
    sending (advertising) to the neighbor before
    output policy rules have been applied.";
  uses ipv6-adj-rib-common;
  uses clear-routes {
    description
      "Clears the adj-rib-out state for the
      containing neighbor. Subsequently, neighbors
      will announce BGP updates to resynchronize
      these routes.";
  }
}
container adj-rib-out-post {
  description
    "Per-neighbor table containing paths eligible for
    sending (advertising) to the neighbor after
    output policy rules have been applied.";
```

```

        uses ipv6-adj-rib-common;
        uses clear-routes {
            description
                "Clears the adj-rib-out state for the
                 containing neighbor. Subsequently, neighbors
                 will announce BGP updates to resynchronize
                 these routes.";
        }
    }
}

description
    "Top level container for BGP RIB.";
}
}

<CODE ENDS>

```

```
<CODE BEGINS> file "ietf-bgp-rib-types@2022-03-06.yang"
submodule ietf-bgp-rib-types {
  yang-version 1.1;
  belongs-to ietf-bgp {
    prefix br;
  }
}
```

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

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

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

Copyright (c) 2021 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to

the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>).

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

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

```
revision 2022-03-06 {
  description
    "Initial Version";
  reference
    "RFC XXXX, BGP Model for Service Provider Network.";
}

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

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

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

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

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

identity bgp-not-selected-bestpath {
  description
    "Base identity for indicating reason a route was was not
    selected by BGP route selection algorithm";
  reference
    "RFC 4271 - Section 9.1";
}

identity local-pref-lower {
  base bgp-not-selected-bestpath;
  description
    "Route has a lower localpref attribute than current best path";
  reference
    "RFC 4271 - Section 9.1.2";
}

identity as-path-longer {
  base bgp-not-selected-bestpath;
  description
    "Route has a longer AS path attribute than current best path";
  reference
    "RFC 4271 - Section 9.1.2.2 (a)";
}

identity origin-type-higher {
  base bgp-not-selected-bestpath;
  description
    "Route has a higher origin type, i.e., IGP origin is preferred
    over EGP or incomplete";
  reference
    "RFC 4271 - Section 9.1.2.2 (b)";
}

identity med-higher {
  base bgp-not-selected-bestpath;
  description
    "Route has a higher MED, or metric, attribute than the current
    best path";
  reference
    "RFC 4271 - Section 9.1.2.2 (c)";
}
```

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

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

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

identity higher-peer-address {
  base bgp-not-selected-bestpath;
  description
    "Route was sent by a peer with a higher IP address";
  reference
    "RFC 4271 - Section 9.1.2.2 (g)";
}

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

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

```
<CODE BEGINS> file "ietf-bgp-rib-attributes@2022-03-06.yang"
submodule ietf-bgp-rib-attributes {
  yang-version 1.1;
  belongs-to ietf-bgp {
    prefix br;
  }

  // import some basic types

  import ietf-bgp-types {
    prefix bgpt;
  }
  import ietf-inet-types {
    prefix inet;
  }
  include ietf-bgp-rib-types;

  // meta

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

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

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

    Copyright (c) 2021 IETF Trust and the persons identified as
    authors of the code. All rights reserved.

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

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

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

```
revision 2022-03-06 {
  description
    "Initial version";
  reference
    "RFC XXXX: BGP YANG Model for Service Provider Network";
}

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

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

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

grouping bgp-unknown-attr-top {
  description
    "Unknown path attributes that are not expected to be shared
```



```
    across route entries, common to LOC-RIB and Adj-RIB";
  container unknown-attributes {
    description
      "Unknown path attributes that were received in the UPDATE
      message which contained the prefix.";

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

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

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

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

```
    leaf extended {
        type boolean;
        description
            "Defines whether the attribute length is one octet
             (if set to false) or two octets (if set to true). Set in
             the fourth high-order bit of the BGP attribute flags
             octet.";
        reference
            "RFC 4271 - A Border Gateway Protocol 4 (BGP-4)";
    }

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

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

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

grouping bgp-adj-rib-attr-state {
    description
```

```
    "Path attributes that are not expected to be shared across
      route entries, specific to Adj-RIB";
  leaf path-id {
    type uint32;
    description
      "When the BGP speaker supports advertisement of multiple
        paths for a prefix, the path identifier is used to
        uniquely identify a route based on the combination of the
        prefix and path id. In the Adj-RIB-In, the path-id value is
        the value received in the update message. In the Loc-RIB,
        if used, it should represent a locally generated path-id
        value for the corresponding route. In Adj-RIB-Out, it
        should be the value sent to a neighbor when add-paths is
        used, i.e., the capability has been negotiated.";
    reference
      "RFC 7911: Advertisement of Multiple Paths in BGP";
  }
}
}
}
<CODE ENDS>

<CODE BEGINS> file "ietf-bgp-rib-tables@2022-03-06.yang"
submodule ietf-bgp-rib-tables {
  yang-version 1.1;
  belongs-to ietf-bgp {
    prefix br;
  }

  // import some basic types

  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types.";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types.";
  }
  import ietf-routing {
    prefix rt;
    reference
      "RFC 8022: A YANG Data Model for Routing Management.";
  }
  import ietf-bgp-types {
    prefix bt;
    reference
```

```
    "RFC XXXX: BGP YANG Model for Service Provider Network.";
}
include ietf-bgp-rib-attributes;

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

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

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

  Copyright (c) 2021 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

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

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

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

```
revision 2022-03-06 {
  description
    "Initial Version";
  reference
    "RFC XXXX, BGP YANG Model for Service Provider Network.";
}

grouping bgp-common-route-annotations-state {
  description
```

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

      The value is the timestamp in seconds relative to
      the Unix Epoch (Jan 1, 1970 00:00:00 UTC).";
  }
  leaf eligible-route {
    type boolean;
    description
      "Indicates that the route is eligible for selection for the
      best route in the Loc-Rib in BGP's Decision Process.";
    reference
      "RFC 4271, Section 9.1.";
  }
  leaf ineligible-reason {
    type identityref {
      base ineligible-route-reason;
    }
    description
      "If the route is ineligible for selection for the best route
      in the Loc-Rib in BGP's Decision process, this indicates the
      reason.";
    reference
      "RFC 4271, Section 9.1.";
  }
}

grouping bgp-adj-rib-in-post-route-annotations-state {
  description
    "Data definitions for information attached to routes in the
    Adj-RIB-in post-policy table";
  leaf best-path {
    type boolean;
    description
      "Current path was selected as the best path.";
  }
}

grouping rib-ext-route-annotations {
  description
    "Extended annotations for routes in the routing tables";
  leaf reject-reason {
    type union {
      type identityref {
```

```
        base bgp-not-selected-bestpath;
    }
    type identityref {
        base bgp-not-selected-policy;
    }
}
description
    "Indicates the reason the route is not used, either due to
    policy filtering or bestpath selection";
}
}

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

grouping bgp-loc-rib-common-attr-refs {
    description
        "Definitions of common references to attribute sets for
        multiple AFI-SAFIs for LOC-RIB tables.";
```

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

grouping bgp-loc-rib-common-keys {
  description
    "Common references used in keys for IPv4 and IPv6
    LOC-RIB entries.";
  leaf origin {
    type union {
      type inet:ip-address;
      type identityref {
        base rt:routing-protocol;
      }
    }
    description
      "Indicates the origin of the route. If the route is learned
      from a neighbor, this value is the neighbor address. If
      the route was injected or redistributed from another
      protocol, the origin indicates the source protocol for the
      route.";
  }
  leaf path-id {
    type uint32;
  }
}
```

```
description
  "If the route is learned from a neighbor, the path-id
  corresponds to the path-id for the route in the
  corresponding adj-rib-in-post table. If the route is
  injected from another protocol, or the neighbor does not
  support BGP add-paths, the path-id should be set
  to zero, also the default value.

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

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

grouping ipv4-adj-rib-common {
  description
    "Common structural grouping for each IPv4 adj-RIB table.";
  container routes {
    config false;
    description
      "Enclosing container for list of routes in the routing
```



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

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

```
        uses rib-ext-route-annotations;
    }
}

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

grouping ipv6-adj-rib-in-post {
    description
        "Common structural grouping for the IPv6 adj-rib-in
        post-policy table.";
    container routes {
        config false;
        description
            "Enclosing container for list of routes in the routing
            table.";
        list route {
            key "prefix path-id";
            description
                "List of routes in the table.";
            leaf prefix {
                type inet:ipv6-prefix;
                description
                    "Prefix for the route.";
            }
        }
    }
}
```

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

8. Contributors

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

9. Acknowledgements

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

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

Special thanks to Robert Wilton who helped convert the YANG models to a NMDA compatible model.

10. References

10.1. Normative references

- [RFC1997] Chandra, R., Traina, P., and T. Li, "BGP Communities Attribute", RFC 1997, DOI 10.17487/RFC1997, August 1996, <<https://www.rfc-editor.org/info/rfc1997>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

- [RFC2439] Villamizar, C., Chandra, R., and R. Govindan, "BGP Route Flap Damping", RFC 2439, DOI 10.17487/RFC2439, November 1998, <<https://www.rfc-editor.org/info/rfc2439>>.
- [RFC2918] Chen, E., "Route Refresh Capability for BGP-4", RFC 2918, DOI 10.17487/RFC2918, September 2000, <<https://www.rfc-editor.org/info/rfc2918>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <<https://www.rfc-editor.org/info/rfc4364>>.
- [RFC4451] McPherson, D. and V. Gill, "BGP MULTI_EXIT_DISC (MED) Considerations", RFC 4451, DOI 10.17487/RFC4451, March 2006, <<https://www.rfc-editor.org/info/rfc4451>>.
- [RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <<https://www.rfc-editor.org/info/rfc4456>>.
- [RFC4659] De Clercq, J., Ooms, D., Carugi, M., and F. Le Faucheur, "BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN", RFC 4659, DOI 10.17487/RFC4659, September 2006, <<https://www.rfc-editor.org/info/rfc4659>>.
- [RFC4724] Sangli, S., Chen, E., Fernando, R., Scudder, J., and Y. Rekhter, "Graceful Restart Mechanism for BGP", RFC 4724, DOI 10.17487/RFC4724, January 2007, <<https://www.rfc-editor.org/info/rfc4724>>.
- [RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <<https://www.rfc-editor.org/info/rfc4760>>.

- [RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <<https://www.rfc-editor.org/info/rfc4761>>.
- [RFC5065] Traina, P., McPherson, D., and J. Scudder, "Autonomous System Confederations for BGP", RFC 5065, DOI 10.17487/RFC5065, August 2007, <<https://www.rfc-editor.org/info/rfc5065>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.
- [RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, DOI 10.17487/RFC5881, June 2010, <<https://www.rfc-editor.org/info/rfc5881>>.
- [RFC5883] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for Multihop Paths", RFC 5883, DOI 10.17487/RFC5883, June 2010, <<https://www.rfc-editor.org/info/rfc5883>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6514] Aggarwal, R., Rosen, E., Morin, T., and Y. Rekhter, "BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs", RFC 6514, DOI 10.17487/RFC6514, February 2012, <<https://www.rfc-editor.org/info/rfc6514>>.
- [RFC6793] Vohra, Q. and E. Chen, "BGP Support for Four-Octet Autonomous System (AS) Number Space", RFC 6793, DOI 10.17487/RFC6793, December 2012, <<https://www.rfc-editor.org/info/rfc6793>>.

- [RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", RFC 6811, DOI 10.17487/RFC6811, January 2013, <<https://www.rfc-editor.org/info/rfc6811>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7911] Walton, D., Retana, A., Chen, E., and J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, DOI 10.17487/RFC7911, July 2016, <<https://www.rfc-editor.org/info/rfc7911>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8177] Lindem, A., Ed., Qu, Y., Yeung, D., Chen, I., and J. Zhang, "YANG Data Model for Key Chains", RFC 8177, DOI 10.17487/RFC8177, June 2017, <<https://www.rfc-editor.org/info/rfc8177>>.
- [RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017, <<https://www.rfc-editor.org/info/rfc8277>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

- [RFC8528] Bjorklund, M. and L. Lhotka, "YANG Schema Mount", RFC 8528, DOI 10.17487/RFC8528, March 2019, <<https://www.rfc-editor.org/info/rfc8528>>.
- [RFC8529] Berger, L., Hopps, C., Lindem, A., Bogdanovic, D., and X. Liu, "YANG Data Model for Network Instances", RFC 8529, DOI 10.17487/RFC8529, March 2019, <<https://www.rfc-editor.org/info/rfc8529>>.
- [RFC9067] Qu, Y., Tantsura, J., Lindem, A., and X. Liu, "A YANG Data Model for Routing Policy", RFC 9067, DOI 10.17487/RFC9067, October 2021, <<https://www.rfc-editor.org/info/rfc9067>>.
- [RFC9127] Rahman, R., Ed., Zheng, L., Ed., Jethanandani, M., Ed., Pallagatti, S., and G. Mirsky, "YANG Data Model for Bidirectional Forwarding Detection (BFD)", RFC 9127, DOI 10.17487/RFC9127, October 2021, <<https://www.rfc-editor.org/info/rfc9127>>.
- [I-D.ietf-tcpm-yang-tcp] Scharf, M., Jethanandani, M., and V. Murgai, "A YANG Model for Transmission Control Protocol (TCP) Configuration", Work in Progress, Internet-Draft, draft-ietf-tcpm-yang-tcp-06, 3 February 2022, <<https://www.ietf.org/archive/id/draft-ietf-tcpm-yang-tcp-06.txt>>.

10.2. Informative references

- [RFC3765] Huston, G., "NOPEER Community for Border Gateway Protocol (BGP) Route Scope Control", RFC 3765, DOI 10.17487/RFC3765, April 2004, <<https://www.rfc-editor.org/info/rfc3765>>.
- [RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <<https://www.rfc-editor.org/info/rfc5082>>.
- [RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <<https://www.rfc-editor.org/info/rfc5925>>.
- [RFC7454] Durand, J., Pepelnjak, I., and G. Doering, "BGP Operations and Security", BCP 194, RFC 7454, DOI 10.17487/RFC7454, February 2015, <<https://www.rfc-editor.org/info/rfc7454>>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.

Appendix A. Examples

This section tries to show some examples in how the model can be used.

A.1. Creating BGP Instance

This example shows how to enable BGP for a IPv4 unicast address family.

[note: '\ ' line wrapping for formatting only]

```
<?xml version="1.0" encoding="UTF-8"?>
<routing
  xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
  <control-plane-protocols>
    <control-plane-protocol>
      <type
        xmlns:bgp="urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:</\
type>
      <name>BGP</name>
      <bgp
        xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
        <global>
          <as>64496</as>
          <afi-safis>
            <afi-safi>
              <name
                xmlns:bt=
                "urn:ietf:params:xml:ns:yang:ietf-bgp-types">bt:ip\
v4-unicast</name>
              </afi-safi>
            </afi-safis>
          </global>
        </bgp>
      </control-plane-protocol>
    </control-plane-protocols>
  </routing>
```


A.2. Neighbor Address Family Configuration

This example shows how to configure a BGP neighbor, where the remote address is 192.0.2.1, the remote AS number is 64497, and the address family of the neighbor is IPv4 unicast. The neighbor is configured for route flap prevention and it set up for standard and large communities. In addition, BFD is configured at a neighbor level with a local multiplier of 2, a desired minimum transmit interval, and a required minimum receive interval of 3.3 ms.

[note: '\ ' line wrapping for formatting only]

```
<!--
  This example shows a neighbor configuration with damping.
-->

<?xml version="1.0" encoding="UTF-8"?>
<routing
  xmlns="urn:ietf:params:xml:ns:yang:ietf-routing"
  xmlns:bt="urn:ietf:params:xml:ns:yang:ietf-bgp-types">
  <control-plane-protocols>
    <control-plane-protocol>
      <type
        xmlns:bgp="urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</\
type>
        <name>name:BGP</name>
        <bgp
          xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
          <global>
            <as>64496</as>
            <afi-safis>
              <afi-safi>
                <name>bt:ipv4-unicast</name>
              </afi-safi>
            </afi-safis>
          </global>
          <neighbors>
            <neighbor>
              <remote-address>192.0.2.1</remote-address>
              <peer-as>64497</peer-as>
              <route-flap-damping>
                <enable>true</enable>
                <suppress-above>4.0</suppress-above>
                <reuse-above>3.0</reuse-above>
                <max-flap>15.0</max-flap>
                <reach-decay>100</reach-decay>
                <unreach-decay>500</unreach-decay>
                <keep-history>1000</keep-history>
```

```

    </route-flap-damping>
    <send-community>bt:standard</send-community>
    <send-community>bt:large</send-community>
    <description>"Peer Router B"</description>
    <afi-safis>
      <afi-safi>
        <name>bt:ipv4-unicast</name>
      </afi-safi>
    </afi-safis>
    <bfd>
      <enabled>true</enabled>
      <local-multiplier>2</local-multiplier>
      <desired-min-tx-interval>3300</desired-min-tx-interval>
    >
      <required-min-rx-interval>3300</required-min-rx-interv\
al>
    </bfd>
  </neighbor>
</neighbors>
</bgp>
</control-plane-protocol>
</control-plane-protocols>
</routing>

```

A.3. IPv6 Neighbor Configuration

This example shows how to configure a BGP peer, where the remote peer has a IPv6 address, uses TCP-AO to secure the session with the peer, and uses non-default timers for hold-time and keepalive.

[note: '\ ' line wrapping for formatting only]

```

<?xml version="1.0" encoding="UTF-8"?>
<key-chains
  xmlns="urn:ietf:params:xml:ns:yang:ietf-key-chain">
  <key-chain>
    <name>bgp-key-chain</name>
  </key-chain>
</key-chains>
<routing
  xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
  <control-plane-protocols>
    <control-plane-protocol>
      <type
        xmlns:bgp="urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</\
type>
      <name>name:BGP</name>
      <bgp

```

```

        xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
    <global>
        <as>64496</as>
        <afi-safis>
            <afi-safi>
                <name
                    xmlns:bt=
                        "urn:ietf:params:xml:ns:yang:ietf-bgp-types">bt:ip\
v6-unicast</name>
                </afi-safi>
            </afi-safis>
        </global>
        <neighbors>
            <neighbor>
                <remote-address>2001:db8::</remote-address>
                <enabled>true</enabled>
                <secure-session-enable>true</secure-session-enable>
                <secure-session>
                    <enable-ao>true</enable-ao>
                    <ao-keychain>bgp-key-chain</ao-keychain>
                </secure-session>
                <peer-as>64497</peer-as>
                <description>"Peer Router B"</description>
                <timers>
                    <hold-time>120</hold-time>
                    <keepalive>70</keepalive>
                </timers>
            <afi-safis>
                <afi-safi>
                    <name
                        xmlns:bt=
                            "urn:ietf:params:xml:ns:yang:ietf-bgp-types">bt:\
ipv6-unicast</name>
                    </afi-safi>
                </afi-safis>
            </neighbor>
        </neighbors>
    </bgp>
</control-plane-protocol>
</control-plane-protocols>
</routing>

```

A.4. VRF Configuration

This example shows how BGP can be configured for two VRFs, red and blue. In this case, the two network instances share a common AS, and distinguish between the instances using the router id.

[note: '\ ' line wrapping for formatting only]

```
<?xml version="1.0" encoding="UTF-8"?>
<network-instances
  xmlns="urn:ietf:params:xml:ns:yang:ietf-network-instance">
  <network-instance>
    <name>vrf-red</name>
    <vrf-root>
      <routing
        xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
        <router-id>192.0.2.1</router-id>
        <control-plane-protocols>
          <control-plane-protocol>
            <type
              xmlns:bgp=
                "urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</type\
>
              <name>BGP</name>
              <bgp
                xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
                <global>
                  <as>64496</as>
                  <afi-safis>
                    <afi-safi>
                      <name
                        xmlns:bt=
                          "urn:ietf:params:xml:ns:yang:ietf-bgp-types"\
>bt:ipv4-unicast</name>
                      </afi-safi>
                    </afi-safis>
                  </global>
                </bgp>
              </control-plane-protocol>
            </control-plane-protocols>
          </routing>
        </vrf-root>
      </network-instance>
    <network-instance>
      <name>vrf-blue</name>
      <vrf-root>
        <routing
          xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
          <router-id>192.0.2.2</router-id>
          <control-plane-protocols>
            <control-plane-protocol>
              <type
                xmlns:bgp=
                  "urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</type\
```

```

>
    <name>BGP</name>
    <bgp
      xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
      <global>
        <as>64496</as>
        <afi-safis>
          <afi-safi>
            <name
              xmlns:bt=
                "urn:ietf:params:xml:ns:yang:ietf-bgp-types"\
>bt:ipv4-unicast</name>
              </afi-safi>
            </afi-safis>
          </global>
        </bgp>
      </control-plane-protocol>
    </control-plane-protocols>
  </routing>
</vrf-root>
</network-instance>
</network-instances>

```

A.5. BGP Policy

Routing policy using community value involves configuring rules to match community values in the inbound or outbound direction. In this example, which is heavily borrowed from the example on the Cisco community page, we look at "match community exact" match, which happens only when BGP updates have the same community values as specified in the community list.

The topology in this example consists of three routers, R1, R2, and R3, configured with AS value of 1, 2 and 3 respectively. R1 advertises 5 prefixes to R2 and R3, as shown below.

- * 1.1.1.1/32 and 2.2.2.2/32 with community 11:11
- * 3.3.3.3/32 and 4.4.4.4/32 with community 11:11 and 22:22
- * 5.5.5.5/32 with community 33:33

Route Policy TO_R2 defines the policy that R1 uses in route updates towards R2. It consists of three statements, statement 10 that has an exact match rule for the prefix list L0andL1, and a set-community action of add for 11:11. The second statement, statement 20, consists of an exact match rule for prefix list L2andL3, with a set community action of remove for 11:11 22:22. The final statement, statement 30, consists of an exact match rule for prefix list L4, with a set community action of replace for 33:33.

[note: '\' line wrapping for formatting only]

```
<?xml version="1.0" encoding="UTF-8"?>
<routing-policy
  xmlns="urn:ietf:params:xml:ns:yang:ietf-routing-policy">
  <defined-sets>
    <prefix-sets>
      <prefix-set>
        <name>L0andL1</name>
        <mode>ipv4</mode>
        <prefixes>
          <prefix-list>
            <ip-prefix>1.1.1.1/32</ip-prefix>
            <mask-length-lower>32</mask-length-lower>
            <mask-length-upper>32</mask-length-upper>
          </prefix-list>
          <prefix-list>
            <ip-prefix>2.2.2.2/32</ip-prefix>
            <mask-length-lower>32</mask-length-lower>
            <mask-length-upper>32</mask-length-upper>
          </prefix-list>
        </prefixes>
      </prefix-set>
      <prefix-set>
        <name>L2andL3</name>
        <mode>ipv4</mode>
        <prefixes>
          <prefix-list>
            <ip-prefix>3.3.3.3/32</ip-prefix>
            <mask-length-lower>32</mask-length-lower>
            <mask-length-upper>32</mask-length-upper>
          </prefix-list>
          <prefix-list>
            <ip-prefix>4.4.4.4/32</ip-prefix>
            <mask-length-lower>32</mask-length-lower>
            <mask-length-upper>32</mask-length-upper>
          </prefix-list>
        </prefixes>
      </prefix-set>
```

```
<prefix-set>
  <name>L4</name>
  <mode>ipv4</mode>
  <prefixes>
    <prefix-list>
      <ip-prefix>5.5.5.5/32</ip-prefix>
      <mask-length-lower>32</mask-length-lower>
      <mask-length-upper>32</mask-length-upper>
    </prefix-list>
  </prefixes>
</prefix-set>
</prefix-sets>
</defined-sets>
<policy-definitions>
  <policy-definition>
    <name>TO_R2</name>
    <statements>
      <statement>
        <name>10</name>
        <conditions>
          <match-prefix-set>
            <prefix-set>L0andL1</prefix-set>
          </match-prefix-set>
        </conditions>
        <actions>
          <bgp-actions
            xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp-policy">\
            <set-community>
              <options>add</options>
              <communities>11:11</communities>
            </set-community>
          </bgp-actions>
        </actions>
      </statement>
      <statement>
        <name>20</name>
        <conditions>
          <match-prefix-set>
            <prefix-set>L2andL3</prefix-set>
          </match-prefix-set>
        </conditions>
        <actions>
          <bgp-actions
            xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp-policy">\
            <set-community>
              <options>remove</options>
```

```

        <communities>11:11</communities>
        <communities>22:22</communities>
      </set-community>
    </bgp-actions>
  </actions>
</statement>
<statement>
  <name>30</name>
  <conditions>
    <match-prefix-set>
      <prefix-set>L4</prefix-set>
    </match-prefix-set>
  </conditions>
  <actions>
    <bgp-actions
      xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp-policy">\
      <set-community>
        <options>replace</options>
        <communities>33:33</communities>
      </set-community>
    </bgp-actions>
  </actions>
</statement>
</statements>
</policy-definition>
</policy-definitions>
</routing-policy>

<routing
  xmlns="urn:ietf:params:xml:ns:yang:ietf-routing"
  xmlns:bt="urn:ietf:params:xml:ns:yang:ietf-bgp-types">
  <control-plane-protocols>
    <control-plane-protocol>
      <type
        xmlns:bgp="urn:ietf:params:xml:ns:yang:ietf-bgp">bgp:bgp</\
type>
      <name>BGP</name>
      <bgp
        xmlns="urn:ietf:params:xml:ns:yang:ietf-bgp">
        <global>
          <as>1</as>
          <afi-safis>
            <afi-safi>
              <name>bt:ipv4-unicast</name>
            </afi-safi>
          </afi-safis>
        </global>

```



```

    <neighbors>
      <neighbor>
        <remote-address>10.1.1.2</remote-address>
        <peer-as>2</peer-as>
        <afi-safis>
          <afi-safi>
            <name>bt:ipv4-unicast</name>
            </afi-safi>
          </afi-safis>
          <send-community>bt:standard</send-community>
          <apply-policy>
            <export-policy>TO_R2</export-policy>
            <default-export-policy>accept-route</default-export-po\
licy>
          </apply-policy>
        </neighbor>
      </neighbors>
    </bgp>
  </control-plane-protocol>
</control-plane-protocols>
</routing>

```

Appendix B. How to add a new AFI and Augment a Module

This section explains how a new AFI can be defined in a new module and how that module can then be augmented. Assume that the new AFI being defined is called 'foo' which extends the base identity of 'afi-safi-type', and the augmentation is to add a new container for 'foo' under two different XPaths. The example shows how the base identity can be extended to add this new AFI, and then use the augmented containers be used to add 'foo' specific information.

```

module example-newafi-bgp {
  yang-version 1.1;
  namespace "http://example.com/ns/example-newafi-bgp";
  prefix example-newafi-bgp;

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

  import ietf-routing {
    prefix rt;
    reference
      "RFC 8349, A YANG Data Model for Routing Management
      (NMDA Version)";
  }

```

```
}

import ietf-bgp {
  prefix "bgp";
  reference
    "RFC XXXX: BGP YANG module for Service Provider Network.";
}

import ietf-bgp-types {
  prefix "bt";
}

organization
  "Newafi model group.";

contact
  "abc@newafi.com";
description
  "This YANG module defines and uses new AFI.";

revision 2022-03-06 {
  description
    "Creating new AFI and using in this model";

  reference
    "RFC XXXX: BGP YANG Model for Service Provider Network.";
}

identity foo {
  base bt:afi-safi-type;
  description
    "New AFI type foo.";
}

augment "/rt:routing/rt:control-plane-protocols/" +
  "rt:control-plane-protocol/bgp:bgp/bgp:global/" +
  "bgp:afi-safis/bgp:afi-safi" {
  when "derived-from-or-self(bgp:name, 'foo')" {
    description
      "This augmentation is valid for a AFI/SAFI instance
        of 'foo'";
  }
  container foo {
    description
      "Container to add 'foo' specific AFI/SAFI information.
        First add the common stuff.";
    uses bgp:mp-all-afi-safi-common;
  }
}
```

```
}

augment "/rt:routing/rt:control-plane-protocols/" +
    "rt:control-plane-protocol/bgp:bgp/" +
    "bgp:rib/bgp:afi-safis/bgp:afi-safi" {
    when "derived-from-or-self(bgp:name, 'foo')" {
        description
            "This augmentation is valid for a AFI/SAFI instance
            of 'foo'";
    }

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

```
container neighbors {
  config false;
  description
    "Enclosing container for neighbor list.";
  list neighbor {
    key "neighbor-address";
    description
      "List of neighbors (peers) of the local BGP speaker.";
    leaf neighbor-address {
      type inet:ip-address;
      description
        "IP address of the BGP neighbor or peer.";
    }
    container adj-rib-in-pre {
      description
        "Per-neighbor table containing the NLRI updates
        received from the neighbor before any local input
        policy rules or filters have been applied. This can
        be considered the 'raw' updates from the neighbor.";
      uses bgp:ipv4-adj-rib-common;
    }
    container adj-rib-in-post {
      description
        "Per-neighbor table containing the paths received from
        the neighbor that are eligible for best-path selection
        after local input policy rules have been applied.";
      uses bgp:ipv4-adj-rib-in-post;
    }
    container adj-rib-out-pre {
      description
        "Per-neighbor table containing paths eligible for
        sending (advertising) to the neighbor before output
        policy rules have been applied.";
      uses bgp:ipv4-adj-rib-common;
    }
    container adj-rib-out-post {
      description
        "Per-neighbor table containing paths eligible for
        sending (advertising) to the neighbor after output
        policy rules have been applied.";
      uses bgp:ipv4-adj-rib-common;
    }
  }
}
```

Appendix C. How to deviate a module

This example shows how the BGP can be deviated to indicate two nodes that the particular implementation is choosing not to support.

```
module example-newco-bgp {
  yang-version 1.1;
  namespace "http://example.com/ns/example-newco-bgp";
  prefix example-newco-bgp;

  import ietf-bgp {
    prefix "bgp";
  }

  organization
    "Newco model group.";

  contact
    "abc@newco.com";
  description
    "This YANG module deviates IETF BGP YANG module.";

  revision 2022-03-06 {
    description
      "Creating NewCo deviations to ietf-bgp model";

    reference
      "RFC XXXX: BGP YANG module for Service Provider Network.";
  }

  deviation "/bgp:bgp/bgp:global/bgp:graceful-restart/" +
    "bgp:restart-time" {
    deviate not-supported;
  }

  deviation "/bgp:bgp/bgp:global/bgp:graceful-restart/" +
    "bgp:stale-route-time" {
    deviate not-supported;
  }
}
```

Appendix D. Complete configuration tree diagram

Here is a complete tree diagram for the configuration and operational part of the model.

```
module: ietf-bgp
```

```
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol:
    +--rw bgp
      +--rw global!
        +--rw as inet:as-number
        +--rw identifier? yang:dotted-quad
        +--rw distance
          +--rw external? uint8
          +--rw internal? uint8
        +--rw confederation
          +--rw enabled? boolean
          +--rw identifier? inet:as-number
          +--rw member-as* inet:as-number
        +--rw graceful-restart {bt:graceful-restart}?
          +--rw enabled? boolean
          +--rw restart-time? uint16
          +--rw stale-routes-time? uint32
          +--rw helper-only? boolean
        +--rw use-multiple-paths
          +--rw enabled? boolean
          +--rw ebgp
            +--rw allow-multiple-as? boolean
            +--rw maximum-paths? uint32
          +--rw ibgp
            +--rw maximum-paths? uint32
        +--rw route-selection-options
          +--rw always-compare-med? boolean
          +--rw ignore-as-path-length? boolean
          +--rw external-compare-router-id? boolean
          +--rw advertise-inactive-routes? boolean
          +--rw enable-aigp? boolean
          +--rw ignore-next-hop-igp-metric? boolean
          +--rw enable-med? boolean
          +--rw med-plus-igp
            +--rw enabled? boolean
            +--rw igp-multiplier? uint16
            +--rw med-multiplier? uint16
        +--rw afi-safis
          +--rw afi-safi* [name]
            +--rw name identityref
            +--rw enabled? boolean
            +--ro total-paths? uint32
            +--ro total-prefixes? uint32
            +--rw graceful-restart {bt:graceful-restart}?
              +--rw enabled? boolean
            +--rw route-selection-options
```

```

+--rw always-compare-med?          boolean
+--rw ignore-as-path-length?       boolean
+--rw external-compare-router-id?  boolean
+--rw advertise-inactive-routes?   boolean
+--rw enable-aigp?                 boolean
+--rw ignore-next-hop-igp-metric?  boolean
+--rw enable-med?                 boolean
+--rw med-plus-igp
  +--rw enabled?                   boolean
  +--rw igp-multiplier?            uint16
  +--rw med-multiplier?            uint16
+--rw use-multiple-paths
  +--rw enabled?                   boolean
  +--rw ebgp
    +--rw allow-multiple-as?       boolean
    +--rw maximum-paths?           uint32
  +--rw ibgp
    +--rw maximum-paths?           uint32
+--rw apply-policy
  +--rw import-policy*              leafref
  +--rw default-import-policy?      default-policy-type
  +--rw export-policy*              leafref
  +--rw default-export-policy?      default-policy-type
+--rw ipv4-unicast
  +--rw prefix-limit
    +--rw max-prefixes?            uint32
    +--rw shutdown-threshold-pct?
      |                             rt-types:percentage
    +--rw restart-timer?           uint32
  +--rw send-default-route?        boolean
+--rw ipv6-unicast
  +--rw prefix-limit
    +--rw max-prefixes?            uint32
    +--rw shutdown-threshold-pct?
      |                             rt-types:percentage
    +--rw restart-timer?           uint32
  +--rw send-default-route?        boolean
+--rw ipv4-labeled-unicast
  +--rw prefix-limit
    +--rw max-prefixes?            uint32
    +--rw shutdown-threshold-pct?
      |                             rt-types:percentage
    +--rw restart-timer?           uint32
+--rw ipv6-labeled-unicast
  +--rw prefix-limit
    +--rw max-prefixes?            uint32
    +--rw shutdown-threshold-pct?
      |                             rt-types:percentage

```

```

|         +---rw restart-timer?                uint32
+---rw l3vpn-ipv4-unicast
|         +---rw prefix-limit
|         +---rw max-prefixes?                uint32
|         +---rw shutdown-threshold-pct?
|         |         rt-types:percentage
|         +---rw restart-timer?                uint32
+---rw l3vpn-ipv6-unicast
|         +---rw prefix-limit
|         +---rw max-prefixes?                uint32
|         +---rw shutdown-threshold-pct?
|         |         rt-types:percentage
|         +---rw restart-timer?                uint32
+---rw l3vpn-ipv4-multicast
|         +---rw prefix-limit
|         +---rw max-prefixes?                uint32
|         +---rw shutdown-threshold-pct?
|         |         rt-types:percentage
|         +---rw restart-timer?                uint32
+---rw l3vpn-ipv6-multicast
|         +---rw prefix-limit
|         +---rw max-prefixes?                uint32
|         +---rw shutdown-threshold-pct?
|         |         rt-types:percentage
|         +---rw restart-timer?                uint32
+---rw l2vpn-vpls
|         +---rw prefix-limit
|         +---rw max-prefixes?                uint32
|         +---rw shutdown-threshold-pct?
|         |         rt-types:percentage
|         +---rw restart-timer?                uint32
+---rw l2vpn-evpn
|         +---rw prefix-limit
|         +---rw max-prefixes?                uint32
|         +---rw shutdown-threshold-pct?
|         |         rt-types:percentage
|         +---rw restart-timer?                uint32
+---rw apply-policy
|         +---rw import-policy*                leafref
|         +---rw default-import-policy?        default-policy-type
|         +---rw export-policy*                leafref
|         +---rw default-export-policy?        default-policy-type
+---ro total-paths?                uint32
+---ro total-prefixes?            uint32
+---rw neighbors
|         +---rw neighbor* [remote-address]
|         |         +---rw remote-address            inet:ip-address
|         |         +---ro local-address?            inet:ip-address

```



```

+--ro local-port?                inet:port-number
+--ro remote-port?               inet:port-number
+--ro peer-type?                 bt:peer-type
+--rw peer-group?
|   -> ../../../../peer-groups/peer-group/name
+--ro identifier?                yang:dotted-quad
+--rw enabled?                   boolean
+--rw secure-session-enable?     boolean
+--rw secure-session
|   +--rw (option)?
|   |   +--:(ao)
|   |   |   +--rw enable-ao?         boolean
|   |   |   +--rw send-id?           uint8
|   |   |   +--rw recv-id?          uint8
|   |   |   +--rw include-tcp-options? boolean
|   |   |   +--rw accept-ao-mismatch? boolean
|   |   |   +--rw ao-keychain?
|   |   |       key-chain:key-chain-ref
|   |   +--:(md5)
|   |   |   +--rw enable-md5?         boolean
|   |   |   +--rw md5-keychain?
|   |   |       key-chain:key-chain-ref
+--rw ttl-security?             uint8
|   {bt:ttl-security}?
+--rw peer-as?                   inet:as-number
+--rw local-as?                  inet:as-number
+--rw remove-private-as?
|   bt:remove-private-as-option
+--rw route-flap-damping {bt:damping}?
|   +--rw enable?                 boolean
|   +--rw suppress-above?         decimal64
|   +--rw reuse-above?            decimal64
|   +--rw max-flap?               decimal64
|   +--rw reach-decay?            uint32
|   +--rw unreach-decay?          uint32
|   +--rw keep-history?           uint32
+--rw send-community*            identityref
|   {bt:send-communities}?
+--rw description?               string
+--rw timers
|   +--rw connect-retry-interval?  uint16
|   +--rw hold-time?              uint16
|   +--rw keepalive?              uint16
|   +--rw min-as-origination-interval? uint16
|   +--rw min-route-advertisement-interval? uint16
+--rw transport
|   +--rw tcp-mss?                 uint16
|   +--rw mtu-discovery?          boolean

```

```

+--rw passive-mode?          boolean
+--rw local-address?         union
+--rw md5-auth-password?     string
+--rw bfd {bt:bfd}?
  +--rw enabled?              boolean
  +--rw local-multiplier?     multiplier
  +--rw (interval-config-type)?
    +--:(tx-rx-intervals)
      +--rw desired-min-tx-interval?  uint32
      +--rw required-min-rx-interval?  uint32
    +--:(single-interval)
      {single-minimum-interval}?
      +--rw min-interval?            uint32
+--rw graceful-restart {bt:graceful-restart}?
  +--rw enabled?          boolean
  +--rw restart-time?     uint16
  +--rw stale-routes-time? uint32
  +--rw helper-only?      boolean
  +--ro peer-restart-time? uint16
  +--ro peer-restarting?   boolean
  +--ro local-restarting?   boolean
  +--ro mode?               enumeration
+--rw logging-options
  +--rw log-neighbor-state-changes?  boolean
+--rw ebgp-multihop
  +--rw enabled?          boolean
  +--rw multihop-ttl?     uint8
+--rw route-reflector
  +--rw cluster-id?        bt:rr-cluster-id-type
  +--rw no-client-reflect?  boolean
  +--rw client?             boolean
+--rw as-path-options
  +--rw allow-own-as?       uint8
  +--rw replace-peer-as?    boolean
+--rw add-paths {bt:add-paths}?
  +--rw receive?            boolean
  +--rw (send)?
    +--:(max)
      +--rw max?            uint8
    +--:(all)
      +--rw all?            empty
  +--rw eligible-prefix-policy? leafref
+--rw use-multiple-paths
  +--rw enabled?            boolean
  +--rw ebgp
    +--rw allow-multiple-as?  boolean
+--rw apply-policy
  +--rw import-policy*       leafref

```

```

+--rw default-import-policy?  default-policy-type
+--rw export-policy*          leafref
+--rw default-export-policy?  default-policy-type
+--rw afi-safis
+--rw afi-safi* [name]
+--rw name                    identityref
+--rw enabled?                boolean
+--ro active?                 boolean
+--ro prefixes
+--ro received?               uint32
+--ro sent?                   uint32
+--ro installed?              uint32
+--rw graceful-restart {bt:graceful-restart}?
+--rw enabled?                boolean
+--ro received?                boolean
+--ro advertised?              boolean
+--ro local-forwarding-state-preserved?
+--ro forwarding-state-preserved?
+--ro end-of-rib-received?     boolean
+--rw apply-policy
+--rw import-policy*          leafref
+--rw default-import-policy?  default-policy-type
+--rw export-policy*          leafref
+--rw default-export-policy?  default-policy-type
+--rw ipv4-unicast
+--rw prefix-limit
+--rw max-prefixes?           uint32
+--rw shutdown-threshold-pct?
+--rw restart-timer?          uint32
+--rw send-default-route?     boolean
+--rw ipv6-unicast
+--rw prefix-limit
+--rw max-prefixes?           uint32
+--rw shutdown-threshold-pct?
+--rw restart-timer?          uint32
+--rw send-default-route?     boolean
+--rw ipv4-labeled-unicast
+--rw prefix-limit

```

	+--rw max-prefixes?	uint32
	+--rw shutdown-threshold-pct?	
	rt-types:percentage	
	+--rw restart-timer?	uint32
+--rw ipv6-labeled-unicast	+--rw prefix-limit	
	+--rw max-prefixes?	uint32
	+--rw shutdown-threshold-pct?	
	rt-types:percentage	
	+--rw restart-timer?	uint32
+--rw l3vpn-ipv4-unicast	+--rw prefix-limit	
	+--rw max-prefixes?	uint32
	+--rw shutdown-threshold-pct?	
	rt-types:percentage	
	+--rw restart-timer?	uint32
+--rw l3vpn-ipv6-unicast	+--rw prefix-limit	
	+--rw max-prefixes?	uint32
	+--rw shutdown-threshold-pct?	
	rt-types:percentage	
	+--rw restart-timer?	uint32
+--rw l3vpn-ipv4-multicast	+--rw prefix-limit	
	+--rw max-prefixes?	uint32
	+--rw shutdown-threshold-pct?	
	rt-types:percentage	
	+--rw restart-timer?	uint32
+--rw l3vpn-ipv6-multicast	+--rw prefix-limit	
	+--rw max-prefixes?	uint32
	+--rw shutdown-threshold-pct?	
	rt-types:percentage	
	+--rw restart-timer?	uint32
+--rw l2vpn-vpls	+--rw prefix-limit	
	+--rw max-prefixes?	uint32
	+--rw shutdown-threshold-pct?	
	rt-types:percentage	
	+--rw restart-timer?	uint32
+--rw l2vpn-evpn	+--rw prefix-limit	
	+--rw max-prefixes?	uint32
	+--rw shutdown-threshold-pct?	
	rt-types:percentage	
	+--rw restart-timer?	uint32
+--rw use-multiple-paths	+--rw enabled?	boolean

```

|         +---rw ebgp
|         |         +---rw allow-multiple-as?    boolean
+---rw session-state?          enumeration
+---ro last-established?       yang:date-and-time
+---ro negotiated-capabilities* identityref
+---ro negotiated-hold-time?   uint16
+---ro last-error?            binary
+---ro fsm-established-time?   yang:gauge32
+---rw treat-as-withdraw?     boolean
+---ro erroneous-update-messages? uint32
+---rw bfd {bt:bfd}?
|         +---rw enabled?                boolean
|         +---rw local-multiplier?       multiplier
+---rw (interval-config-type)?
|         +---:(tx-rx-intervals)
|         |         +---rw desired-min-tx-interval?    uint32
|         |         +---rw required-min-rx-interval?   uint32
|         +---:(single-interval) {single-minimum-interval}?
|         |         +---rw min-interval?                uint32
+---rw statistics
|         +---ro peer-fsm-established-transitions?
|         |         yang:counter64
+---ro fsm-established-transitions?
|         yang:counter32
+---ro messages
|         +---ro in-total-messages?        yang:counter32
|         +---ro out-total-messages?       yang:counter32
|         +---ro in-update-elapsed-time?   yang:gauge32
+---ro sent
|         +---ro updates-received?        uint64
|         +---ro updates-sent?            uint64
|         +---ro messages-received?       uint64
|         +---ro messages-sent?           uint64
|         +---ro notification?            uint64
+---ro received
|         +---ro updates-received?        uint64
|         +---ro updates-sent?            uint64
|         +---ro messages-received?       uint64
|         +---ro messages-sent?           uint64
|         +---ro notification?            uint64
+---ro queues
|         +---ro input?                    uint32
|         +---ro output?                   uint32
+---x clear {bt:clear-statistics}?
|         +---w input
|         |         +---w clear-at?        yang:date-and-time
+---ro output
|         +---ro clear-finished-at?       yang:date-and-time

```

```

+---n established
|   +--- remote-address?   -> ../../neighbor/remote-address
|   +--- last-error?       -> ../../neighbor/last-error
|   +--- session-state?    -> ../../neighbor/session-state
+---n backward-transition
|   +--- remote-addr?      -> ../../neighbor/remote-address
|   +--- last-error?       -> ../../neighbor/last-error
|   +--- session-state?    -> ../../neighbor/session-state
+---x clear {bt:clear-neighbors}?
|   +---w input
|   |   +---w (operation)?
|   |   |   +---:(operation-admin)
|   |   |   |   +---w admin?           empty
|   |   |   +---:(operation-hard)
|   |   |   |   +---w hard?            empty
|   |   |   +---:(operation-soft)
|   |   |   |   +---w soft?            empty
|   |   |   +---:(operation-soft-inbound)
|   |   |   |   +---w soft-inbound?    empty {bt:route-refresh}?
|   |   +---w clear-at?                yang:date-and-time
|   +---ro output
|   |   +---ro clear-finished-at?      yang:date-and-time
+---rw peer-groups
|   +---rw peer-group* [name]
|   |   +---rw name                    string
|   |   +---rw secure-session-enable?  boolean
|   |   +---rw secure-session
|   |   |   +---rw (option)?
|   |   |   |   +---:(ao)
|   |   |   |   |   +---rw enable-ao?          boolean
|   |   |   |   |   +---rw send-id?            uint8
|   |   |   |   |   +---rw rcv-id?            uint8
|   |   |   |   |   +---rw include-tcp-options? boolean
|   |   |   |   |   +---rw accept-ao-mismatch? boolean
|   |   |   |   |   +---rw ao-keychain?
|   |   |   |   |   |   key-chain:key-chain-ref
|   |   |   |   +---:(md5)
|   |   |   |   |   +---rw enable-md5?          boolean
|   |   |   |   |   +---rw md5-keychain?
|   |   |   |   |   |   key-chain:key-chain-ref
|   |   |   |   +---:(ipsec)
|   |   |   |   |   +---rw sa?                  string
|   |   +---rw ttl-security?              uint8 {bt:ttl-security}?
|   |   +---rw peer-as?                    inet:as-number
|   |   +---rw local-as?                    inet:as-number
|   |   +---rw remove-private-as?
|   |   |   bt:remove-private-as-option
|   +---rw route-flap-damping {bt:damping}?

```

```

+--rw enable?                boolean
+--rw suppress-above?        decimal64
+--rw reuse-above?           decimal64
+--rw max-flap?               decimal64
+--rw reach-decay?            uint32
+--rw unreach-decay?          uint32
+--rw keep-history?           uint32
+--rw send-community*         identityref
|                               {bt:send-communities}?
+--rw description?            string
+--rw timers
|   +--rw connect-retry-interval?    uint16
|   +--rw hold-time?                  uint16
|   +--rw keepalive?                  uint16
|   +--rw min-as-origination-interval? uint16
|   +--rw min-route-advertisement-interval? uint16
+--rw transport
|   +--rw tcp-mss?                    uint16
|   +--rw mtu-discovery?              boolean
|   +--rw passive-mode?               boolean
|   +--rw local-address?              union
|   +--rw md5-auth-password?          string
|   +--rw bfd {bt:bfd}?
|       +--rw enabled?                boolean
|       +--rw local-multiplier?        multiplier
|       +--rw (interval-config-type)?
|           +--:(tx-rx-intervals)
|               +--rw desired-min-tx-interval?    uint32
|               +--rw required-min-rx-interval?    uint32
|           +--:(single-interval)
|               {single-minimum-interval}?
|               +--rw min-interval?                uint32
+--rw graceful-restart {bt:graceful-restart}?
|   +--rw enabled?                boolean
|   +--rw restart-time?            uint16
|   +--rw stale-routes-time?        uint32
|   +--rw helper-only?             boolean
|   +--ro peer-restart-time?        uint16
|   +--ro peer-restarting?          boolean
|   +--ro local-restarting?         boolean
|   +--ro mode?                    enumeration
+--rw logging-options
|   +--rw log-neighbor-state-changes? boolean
+--rw ebgp-multihop
|   +--rw enabled?                boolean
|   +--rw multihop-ttl?            uint8
+--rw route-reflector
|   +--rw cluster-id?              bt:rr-cluster-id-type

```

```

|   +---rw no-client-reflect?    boolean
|   +---rw client?               boolean
+---rw as-path-options
|   +---rw allow-own-as?         uint8
|   +---rw replace-peer-as?     boolean
+---rw add-paths {bt:add-paths}?
|   +---rw receive?             boolean
|   +---rw (send)?
|   |   +---:(max)
|   |   |   +---rw max?         uint8
|   |   +---:(all)
|   |   |   +---rw all?         empty
|   +---rw eligible-prefix-policy? leafref
+---rw use-multiple-paths
|   +---rw enabled?             boolean
|   +---rw ebgp
|   |   +---rw allow-multiple-as? boolean
+---rw apply-policy
|   +---rw import-policy*        leafref
|   +---rw default-import-policy? default-policy-type
|   +---rw export-policy*        leafref
|   +---rw default-export-policy? default-policy-type
+---rw afi-safis
|   +---rw afi-safi* [name]
|   |   +---rw name              identityref
|   |   +---rw enabled?          boolean
|   |   +---rw graceful-restart {bt:graceful-restart}?
|   |   |   +---rw enabled?      boolean
|   |   +---rw use-multiple-paths
|   |   |   +---rw enabled?      boolean
|   |   |   +---rw ebgp
|   |   |   |   +---rw allow-multiple-as? boolean
|   |   +---rw apply-policy
|   |   |   +---rw import-policy*        leafref
|   |   |   +---rw default-import-policy? default-policy-type
|   |   |   |   +---rw export-policy*        leafref
|   |   |   +---rw default-export-policy? default-policy-type
|   +---rw ipv4-unicast
|   |   +---rw prefix-limit
|   |   |   +---rw max-prefixes?          uint32
|   |   |   +---rw shutdown-threshold-pct?
|   |   |   |   rt-types:percentage
|   |   |   +---rw restart-timer?          uint32
|   |   +---rw send-default-route?        boolean
|   +---rw ipv6-unicast
|   |   +---rw prefix-limit

```



```

| | | +--rw max-prefixes?          uint32
| | | +--rw shutdown-threshold-pct?
| | | |   rt-types:percentage
| | | +--rw restart-timer?        uint32
| | +--rw send-default-route?    boolean
+--rw ipv4-labeled-unicast
| | +--rw prefix-limit
| | | +--rw max-prefixes?          uint32
| | | +--rw shutdown-threshold-pct?
| | | |   rt-types:percentage
| | | +--rw restart-timer?        uint32
+--rw ipv6-labeled-unicast
| | +--rw prefix-limit
| | | +--rw max-prefixes?          uint32
| | | +--rw shutdown-threshold-pct?
| | | |   rt-types:percentage
| | | +--rw restart-timer?        uint32
+--rw l3vpn-ipv4-unicast
| | +--rw prefix-limit
| | | +--rw max-prefixes?          uint32
| | | +--rw shutdown-threshold-pct?
| | | |   rt-types:percentage
| | | +--rw restart-timer?        uint32
+--rw l3vpn-ipv6-unicast
| | +--rw prefix-limit
| | | +--rw max-prefixes?          uint32
| | | +--rw shutdown-threshold-pct?
| | | |   rt-types:percentage
| | | +--rw restart-timer?        uint32
+--rw l3vpn-ipv4-multicast
| | +--rw prefix-limit
| | | +--rw max-prefixes?          uint32
| | | +--rw shutdown-threshold-pct?
| | | |   rt-types:percentage
| | | +--rw restart-timer?        uint32
+--rw l3vpn-ipv6-multicast
| | +--rw prefix-limit
| | | +--rw max-prefixes?          uint32
| | | +--rw shutdown-threshold-pct?
| | | |   rt-types:percentage
| | | +--rw restart-timer?        uint32
+--rw l2vpn-vpls
| | +--rw prefix-limit
| | | +--rw max-prefixes?          uint32
| | | +--rw shutdown-threshold-pct?
| | | |   rt-types:percentage
| | | +--rw restart-timer?        uint32
+--rw l2vpn-evpn

```

```

    |             +---rw prefix-limit
    |             +---rw max-prefixes?          uint32
    |             +---rw shutdown-threshold-pct?
    |             |             rt-types:percentage
    |             +---rw restart-timer?          uint32
+---rw interfaces
    |   +---rw interface* [name]
    |   |   +---rw name          if:interface-ref
    |   |   +---rw bfd {bt:bfd}?
    |   |   +---rw enabled?      boolean
+---ro rib
    |   +---ro attr-sets
    |   |   +---ro attr-set* [index]
    |   |   |   +---ro index          uint64
    |   |   |   +---ro attributes
    |   |   |   |   +---ro origin?
    |   |   |   |   |   bt:bgp-origin-attr-type
    |   |   |   |   +---ro atomic-aggregate?      boolean
    |   |   |   |   +---ro next-hop?              inet:ip-address
    |   |   |   |   +---ro link-local-next-hop?    inet:ipv6-address
    |   |   |   |   +---ro med?                    uint32
    |   |   |   |   +---ro local-pref?              uint32
    |   |   |   |   +---ro originator-id?          yang:dotted-quad
    |   |   |   |   +---ro cluster-list*           yang:dotted-quad
    |   |   |   |   +---ro aigp-metric?            uint64
    |   |   |   +---ro aggregator
    |   |   |   |   +---ro as?          inet:as-number
    |   |   |   |   +---ro address?     inet:ipv4-address
    |   |   |   +---ro aggregator4
    |   |   |   |   +---ro as4?         inet:as-number
    |   |   |   |   +---ro address?     inet:ipv4-address
    |   |   |   +---ro as-path
    |   |   |   |   +---ro segment* []
    |   |   |   |   |   +---ro type?     identityref
    |   |   |   |   |   +---ro member*   inet:as-number
    |   |   |   +---ro as4-path
    |   |   |   |   +---ro segment* []
    |   |   |   |   |   +---ro type?     identityref
    |   |   |   |   |   +---ro member*   inet:as-number
    |   |   +---ro communities
    |   |   |   +---ro community* [index]
    |   |   |   |   +---ro index          uint64
    |   |   |   |   +---ro community*     union
    |   |   +---ro ext-communities
    |   |   |   +---ro ext-community* [index]
    |   |   |   |   +---ro index          uint64
    |   |   |   |   +---ro ext-community*  rt:route-target
    |   |   +---ro large-communities

```

```

|   +--ro large-community* [index]
|   |   +--ro index          uint64
|   |   +--ro large-community*  bt:bgp-large-community-type
+--ro afi-safis
|   +--ro afi-safi* [name]
|   |   +--ro name          identityref
|   |   +--ro ipv4-unicast
|   |   |   +--ro loc-rib
|   |   |   |   +--ro routes
|   |   |   |   |   +--ro route* [prefix origin path-id]
|   |   |   |   |   |   +--ro prefix
|   |   |   |   |   |   |   inet:ipv4-prefix
|   |   |   |   |   |   +--ro origin          union
|   |   |   |   |   |   +--ro path-id          uint32
|   |   |   |   |   |   +--ro attr-index?      leafref
|   |   |   |   |   |   +--ro community-index?  leafref
|   |   |   |   |   |   +--ro ext-community-index? leafref
|   |   |   |   |   |   +--ro last-modified?
|   |   |   |   |   |   |   yang:timeticks
|   |   |   |   |   |   +--ro eligible-route?   boolean
|   |   |   |   |   |   +--ro ineligible-reason? identityref
|   |   |   |   |   |   +--ro unknown-attributes
|   |   |   |   |   |   |   +--ro unknown-attribute* [attr-type]
|   |   |   |   |   |   |   |   +--ro optional?   boolean
|   |   |   |   |   |   |   |   +--ro transitive?  boolean
|   |   |   |   |   |   |   |   +--ro partial?    boolean
|   |   |   |   |   |   |   |   +--ro extended?   boolean
|   |   |   |   |   |   |   |   +--ro attr-type    uint8
|   |   |   |   |   |   |   |   +--ro attr-len?   uint16
|   |   |   |   |   |   |   |   +--ro attr-value?  binary
|   |   |   |   |   |   +--ro reject-reason?     union
|   |   +--ro neighbors
|   |   |   +--ro neighbor* [neighbor-address]
|   |   |   |   +--ro neighbor-address  inet:ip-address
|   |   |   |   +--ro adj-rib-in-pre
|   |   |   |   |   +--ro routes
|   |   |   |   |   |   +--ro route* [prefix path-id]
|   |   |   |   |   |   |   +--ro prefix
|   |   |   |   |   |   |   |   inet:ipv4-prefix
|   |   |   |   |   |   |   +--ro path-id          uint32
|   |   |   |   |   |   |   +--ro attr-index?      leafref
|   |   |   |   |   |   |   +--ro community-index?  leafref
|   |   |   |   |   |   |   +--ro ext-community-index? leafref
|   |   |   |   |   |   |   +--ro last-modified?
|   |   |   |   |   |   |   |   yang:timeticks
|   |   |   |   |   |   |   +--ro eligible-route?   boolean
|   |   |   |   |   |   |   |   +--ro ineligible-reason?

```



```

+--ro clear-routes {bt:clear-routes}?
  +---x clear
    +---w input
      +---w clear-at?
        yang:date-and-time
    +--ro output
      +--ro clear-finished-at?
        yang:date-and-time
+--ro adj-rib-out-pre
+--ro routes
  +--ro route* [prefix path-id]
    +--ro prefix
      inet:ipv4-prefix
    +--ro path-id
      uint32
    +--ro attr-index?
      leafref
    +--ro community-index?
      leafref
    +--ro ext-community-index?
      leafref
    +--ro last-modified?
      yang:timeticks
    +--ro eligible-route?
      boolean
    +--ro ineligible-reason?
      identityref
    +--ro unknown-attributes
      +--ro unknown-attribute*
        [attr-type]
          +--ro optional?
            boolean
          +--ro transitive?
            boolean
          +--ro partial?
            boolean
          +--ro extended?
            boolean
          +--ro attr-type
            uint8
          +--ro attr-len?
            uint16
          +--ro attr-value?
            binary
        +--ro reject-reason?
          union
+--ro clear-routes {bt:clear-routes}?
  +---x clear
    +---w input
      +---w clear-at?
        yang:date-and-time
    +--ro output
      +--ro clear-finished-at?
        yang:date-and-time
+--ro adj-rib-out-post
+--ro routes
  +--ro route* [prefix path-id]
    +--ro prefix
      inet:ipv4-prefix
    +--ro path-id
      uint32

```

```

+---ro attr-index?          leafref
+---ro community-index?     leafref
+---ro ext-community-index?  leafref
+---ro last-modified?
|   yang:timeticks
+---ro eligible-route?
|   boolean
+---ro ineligible-reason?
|   identityref
+---ro unknown-attributes
|   +---ro unknown-attribute*
|       [attr-type]
|       +---ro optional?      boolean
|       +---ro transitive?    boolean
|       +---ro partial?       boolean
|       +---ro extended?      boolean
|       +---ro attr-type      uint8
|       +---ro attr-len?      uint16
|       +---ro attr-value?    binary
+---ro reject-reason?        union
+---ro clear-routes {bt:clear-routes}?
|   +---x clear
|   +---w input
|       +---w clear-at?
|           yang:date-and-time
+---ro output
|   +---ro clear-finished-at?
|       yang:date-and-time
+---ro ipv6-unicast
+---ro loc-rib
|   +---ro routes
|       +---ro route* [prefix origin path-id]
|       +---ro prefix
|           inet:ipv6-prefix
|       +---ro origin          union
|       +---ro path-id         uint32
|       +---ro attr-index?     leafref
|       +---ro community-index? leafref
|       +---ro ext-community-index? leafref
|       +---ro last-modified?
|           |   yang:timeticks
|       +---ro eligible-route?    boolean
|       +---ro ineligible-reason? identityref
|       +---ro unknown-attributes
|           |   +---ro unknown-attribute* [attr-type]
|           |       +---ro optional?      boolean
|           |       +---ro transitive?    boolean
|           |       +---ro partial?       boolean

```

```

|         |         +---ro extended?         boolean
|         |         +---ro attr-type          uint8
|         |         +---ro attr-len?          uint16
|         |         +---ro attr-value?       binary
|         |         +---ro reject-reason?     union
+---ro neighbors
  +---ro neighbor* [neighbor-address]
    +---ro neighbor-address    inet:ip-address
    +---ro adj-rib-in-pre
      +---ro routes
        +---ro route* [prefix path-id]
          +---ro prefix
            |          inet:ipv6-prefix
          +---ro path-id          uint32
          +---ro attr-index?      leafref
          +---ro community-index? leafref
          +---ro ext-community-index? leafref
          +---ro last-modified?
            |          yang:timeticks
          +---ro eligible-route?
            |          boolean
          +---ro ineligible-reason?
            |          identityref
          +---ro unknown-attributes
            +---ro unknown-attribute*
              [attr-type]
                +---ro optional?    boolean
                +---ro transitive?   boolean
                +---ro partial?      boolean
                +---ro extended?     boolean
                +---ro attr-type     uint8
                +---ro attr-len?     uint16
                +---ro attr-value?   binary
          +---ro reject-reason?     union
      +---ro clear-routes {bt:clear-routes}?
        +---x clear
          +---w input
            +---w clear-at?
              |          yang:date-and-time
          +---ro output
            +---ro clear-finished-at?
              |          yang:date-and-time
+---ro adj-rib-in-post
  +---ro routes
    +---ro route* [prefix path-id]
      +---ro prefix
        |          inet:ipv6-prefix
      +---ro path-id          uint32

```

```

+--ro attr-index?          leafref
+--ro community-index?     leafref
+--ro ext-community-index? leafref
+--ro last-modified?
|   yang:timeticks
+--ro eligible-route?
|   boolean
+--ro ineligible-reason?
|   identityref
+--ro best-path?
|   boolean
+--ro unknown-attributes
|   +--ro unknown-attribute*
|       [attr-type]
|       +--ro optional?    boolean
|       +--ro transitive?  boolean
|       +--ro partial?     boolean
|       +--ro extended?    boolean
|       +--ro attr-type    uint8
|       +--ro attr-len?    uint16
|       +--ro attr-value?  binary
+--ro reject-reason?       union
+--ro clear-routes {bt:clear-routes}?
|   +---x clear
|   |   +---w input
|   |   |   +---w clear-at?
|   |   |       yang:date-and-time
|   |   +--ro output
|   |       +--ro clear-finished-at?
|   |           yang:date-and-time
+--ro adj-rib-out-pre
+--ro routes
|   +--ro route* [prefix path-id]
|       +--ro prefix
|           inet:ipv6-prefix
|       +--ro path-id          uint32
|       +--ro attr-index?     leafref
|       +--ro community-index? leafref
|       +--ro ext-community-index? leafref
|       +--ro last-modified?
|           |   yang:timeticks
|       +--ro eligible-route?
|           |   boolean
|       +--ro ineligible-reason?
|           |   identityref
|       +--ro unknown-attributes
|           |   +--ro unknown-attribute*
|               |   [attr-type]

```



```

|--ro optional?          boolean
|--ro transitive?       boolean
|--ro partial?          boolean
|--ro extended?         boolean
|--ro attr-type         uint8
|--ro attr-len?         uint16
|--ro attr-value?       binary
+--ro reject-reason?     union
+--ro clear-routes {bt:clear-routes}?
+---x clear
+---w input
|   +---w clear-at?
|       yang:date-and-time
+--ro output
+--ro clear-finished-at?
|   yang:date-and-time
+--ro adj-rib-out-post
+--ro routes
+--ro route* [prefix path-id]
+--ro prefix
|   inet:ipv6-prefix
+--ro path-id            uint32
+--ro attr-index?        leafref
+--ro community-index?   leafref
+--ro ext-community-index? leafref
+--ro last-modified?
|   yang:timeticks
+--ro eligible-route?
|   boolean
+--ro ineligible-reason?
|   identityref
+--ro unknown-attributes
+--ro unknown-attribute*
|   [attr-type]
|       +--ro optional?          boolean
|       +--ro transitive?       boolean
|       +--ro partial?          boolean
|       +--ro extended?         boolean
|       +--ro attr-type         uint8
|       +--ro attr-len?         uint16
|       +--ro attr-value?       binary
+--ro reject-reason?     union
+--ro clear-routes {bt:clear-routes}?
+---x clear
+---w input
|   +---w clear-at?
|       yang:date-and-time
+--ro output

```



```

    |   +---rw lt-or-eq?      empty
    |   +---:(gt-or-eq)
    |   +---rw gt-or-eq?      empty
+---rw as-path-length
    |   +---rw as-path-length?  uint32
    |   +---rw (operation)?
    |   |   +---:(eq)
    |   |   |   +---rw eq?      empty
    |   |   +---:(lt-or-eq)
    |   |   |   +---rw lt-or-eq?  empty
    |   |   +---:(gt-or-eq)
    |   |   |   +---rw gt-or-eq?  empty
+---rw match-community-set
    |   +---rw community-set?    leafref
    |   +---rw match-set-options? match-set-options-type
+---rw match-ext-community-set
    |   +---rw ext-community-set?  leafref
    |   +---rw match-set-options?  match-set-options-type
+---rw match-large-community-set
    |   +---rw ext-community-set?  leafref
    |   +---rw match-set-options?  match-set-options-type
+---rw match-as-path-set
    |   +---rw as-path-set?        leafref
    |   +---rw match-set-options?  match-set-options-type
+---rw match-next-hop-set
    |   +---rw next-hop-set?        leafref
    |   +---rw match-set-options?  match-set-options-type
augment /rt-pol:routing-policy/rt-pol:policy-definitions
    /rt-pol:policy-definition/rt-pol:statements
    /rt-pol:statement/rt-pol:actions:
+---rw bgp-actions
    |   +---rw set-route-origin?    bt:bgp-origin-attr-type
    |   +---rw set-local-pref?      uint32
    |   +---rw set-next-hop?        bgp-next-hop-type
    |   +---rw set-med?             bgp-set-med-type
    |   +---rw set-as-path-prepend
    |   |   +---rw repeat-n?      uint8
+---rw set-community
    |   +---rw options?
    |   |   bgp-set-community-option-type
    |   +---rw (method)?
    |   |   +---:(inline)
    |   |   |   +---rw communities*      union
    |   |   +---:(reference)
    |   |   |   +---rw community-set-ref?  leafref
+---rw set-ext-community
    |   +---rw options?
    |   |   bgp-set-community-option-type

```

```
|   +--rw (method)?
|   |   +--:(inline)
|   |   |   +--rw communities*           rt-types:route-target
|   |   +--:(reference)
|   |   |   +--rw ext-community-set-ref?  leafref
+--rw set-large-community
|   +--rw options?
|   |   bgp-set-community-option-type
+--rw (method)?
|   +--:(inline)
|   |   +--rw communities*
|   |   |   bt:bgp-large-community-type
+--:(reference)
|   +--rw large-community-set-ref?  leafref
```

Authors' Addresses

Mahesh Jethanandani
Kloud Services
Email: mjethanandani@gmail.com

Keyur Patel
Arrcus
CA
United States of America
Email: keyur@arrcus.com

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI 48176
United States of America
Email: shares@ndzh.com

Jeffrey Haas
Juniper Networks
Email: jhaas@pfrc.org

Network Working Group
Internet-Draft
Obsoletes: 6472 (if approved)
Updates: 4271 5065 (if approved)
Intended status: Standards Track
Expires: 8 September 2022

W. Kumari
Google, Inc.
K. Sriram
L. Hannachi
USA NIST
J. Haas
Juniper Networks, Inc.
7 March 2022

Deprecation of AS_SET and AS_CONFED_SET in BGP
draft-ietf-idr-deprecate-as-set-confed-set-07

Abstract

BCP 172 (i.e., RFC 6472) recommends not using AS_SET and AS_CONFED_SET in the Border Gateway Protocol. This document advances this recommendation to a standards requirement in BGP; it proscribes the use of the AS_SET and AS_CONFED_SET types of path segments in the AS_PATH. This is done to simplify the design and implementation of BGP and to make the semantics of the originator of a route clearer. This will also simplify the design, implementation, and deployment of various BGP security mechanisms. This document (if approved) updates RFC 4271 and RFC 5065 by eliminating AS_SET and AS_CONFED_SET types, and obsoletes RFC 6472.

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 8 September 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. Recommendations	3
4. Updates to Existing RFCs	4
5. Operational Considerations	4
6. Security Considerations	5
7. IANA Considerations	5
8. Acknowledgements	5
9. References	5
9.1. Normative References	5
9.2. Informative References	6
Authors' Addresses	7

1. Introduction

BCP 172 [RFC6472] makes a recommendation for not using AS_SET (see [RFC4271]) and AS_CONFED_SET (see [RFC5065]) in the Border Gateway Protocol (BGP). This document advances the BCP recommendation to a standards requirement in BGP; it proscribes the use of the AS_SET and AS_CONFED_SET types of path segments in the AS_PATH.

The AS_SET path segment in the AS_PATH attribute (Sections 4.3 and 5.1.2 of [RFC4271]) is created by a router that is performing route aggregation and contains an unordered set of Autonomous Systems (ASes) that contributing prefixes in the aggregate have traversed. The AS_CONFED_SET path segment (see [RFC5065]) in the AS_PATH attribute is created by a router that is performing route aggregation and contains an unordered set of Member AS Numbers in the local confederation that contributing prefixes in the aggregate have traversed. It is very similar to an AS_SET but is used within a confederation.

By performing aggregation, a router is combining multiple existing routes into a single new route. The aggregation together with the use of AS_SET blurs the semantics of origin AS for the prefix being announced. Therefore, the aggregation with AS_SET (or AS_CONFED_SET) can cause operational issues, such as not being able to authenticate

a route origin for the aggregate prefix in new BGP security technologies such as those that take advantage of X.509 extensions for IP addresses and AS identifiers [RFC3779] [RFC6480] [RFC6811] [RFC8205]. This in turn could result in reachability problems for the aggregated prefix and its components (i.e., more specific prefixes).

From analysis of past Internet routing data, it is apparent that aggregation that involves AS_SETs is very seldom used in practice on the public Internet [Analysis] and when it is used, it is often used incorrectly -- only a single AS in the AS_SET are by far the most common cases. Also, very often the same AS appears in the AS_SEQUENCE and the AS_SET in the BGP update. The occurrence of reserved AS numbers ([IANA-SP-ASN]) is also somewhat frequent. Because the aggregation involving AS_SETs is very rarely used, the reduction in table size provided by this is extremely small, and any advantage thereof is outweighed by additional complexity in BGP. As noted above, AS_SETs also pose impediments to implementation of new BGP security technologies.

2. Requirements Language

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

3. Recommendations

BGP speakers conforming to this document (i.e., conformant BGP speakers) MUST NOT locally generate BGP UPDATE messages containing AS_SET or AS_CONFED_SET. Conformant BGP speakers SHOULD NOT send BGP UPDATE messages containing AS_SET or AS_CONFED_SET. Upon receipt of such messages, conformant BGP speakers SHOULD use the "Treat-as-withdraw" error handling behavior as per [RFC7606].

If a network operator wishes to consider BGP UPDATE messages with AS_SET or AS_CONFED_SET (received from an external peer) for path selection, they MAY have a feature (knob) in their BGP speaker to opt to do so on a per peer basis. The operator should understand the full implications of choosing this option. There is no knob concerning locally generated BGP UPDATE messages, i.e., as stated before a conformant BGP speaker must not locally generate BGP UPDATE messages with AS_SET or AS_CONFED_SET.

Network operators MUST NOT locally generate any new announcements containing AS_SET or AS_CONFED_SET. If they have announced routes with AS_SET or AS_CONFED_SET in them, then they SHOULD withdraw those routes and re-announce routes for the aggregate or component prefixes (i.e., the more specific routes subsumed by the previously aggregated route) without AS_SET or AS_CONFED_SET in the updates.

It is worth noting that new BGP security technologies (such as those that take advantage of X.509 extensions for IP addresses and AS identifiers [RFC3779] [RFC6480] [RFC6811] [RFC8205]) might not support routes with AS_SET or AS_CONFED_SET in them, and may treat routes containing them as infeasible even before the updated BGP in this document is implemented.

4. Updates to Existing RFCs

This document deprecates the AS_SET (type 1) AS_PATH segment type from [RFC4271]. BGP speakers conforming to this document (i.e., conformant BGP speakers) MUST NOT locally generate BGP UPDATE messages containing AS_SET. Conformant BGP speakers SHOULD NOT send BGP UPDATE messages containing AS_SET. Upon receipt of such messages, conformant BGP speakers SHOULD use the "Treat-as-withdraw" error handling behavior as per [RFC7606].

This document deprecates the AS_CONFED_SET (type 4) AS_PATH segment type from [RFC5065]. Conformant BGP speakers MUST NOT locally generate BGP UPDATE messages containing AS_CONFED_SET. Conformant BGP speakers SHOULD NOT send BGP UPDATE messages containing AS_CONFED_SET. Upon receipt of such messages, conformant BGP speakers SHOULD use the "Treat-as-withdraw" error handling behavior as per [RFC7606].

Wherever mentions of AS_SET or AS_CONFED_SET occur in [RFC4271] and [RFC5065], appropriate modification or elimination of the text must be made in future RFCs that would replace these RFCs, consistent with the deprecation of AS_SET and AS_CONFED_SET.

5. Operational Considerations

When aggregating prefixes, network operators MUST use brief aggregation. In brief aggregation, the AGGREGATOR attribute is included but the AS_SET or AS_CONFED_SET attribute is not included.

When doing the above, operators MUST form the aggregate at the border in the outbound BGP policy and omit any prefixes from the AS that the aggregate is being advertised to. In other words, an aggregate prefix MUST NOT be announced to the contributing ASes. Instead, more specific prefixes (from the aggregate) MUST be announced to each

contributing AS, excluding any that were learned from the contributing AS in consideration. For illustration, if p1/24 (from AS1), p2/24 (from AS2), p3/24 (from AS3) and p4/24 (from AS4) are aggregated to p/22, then p/22 will not be announced to AS1, AS2, AS3, or AS4. Instead, as further illustration, p1/24, p2/24 and p4/24 are announced to AS3. Or, possibly q/23 (aggregate of p1/24 and p2/24) and p4/24 are announced to AS3.

Operators MUST install egress filters to block data packets when the destination address belongs to an internal prefix. Similarly, any known single-homed customer prefix MUST also be included in the egress filters except on the interface for that customer. This mitigates looping in the data plane when connection to such an internal or customer prefix is lost. This mechanism effectively compensates for the lack of the additional loop detection capability accorded by AS_SETs (if they were allowed).

6. Security Considerations

This document obsoletes the use of aggregation techniques that create AS_SETs or AS_CONFED_SETs. Obsoleting these path segment types from BGP and removal of the related code from implementations would potentially decrease the attack surface for BGP. Deployments of new BGP security technologies [RFC6480] [RFC6811] [RFC8205] benefit greatly if AS_SET and AS_CONFED_SET are not used in BGP.

7. IANA Considerations

This document requires no IANA actions.

8. Acknowledgements

The authors would like to thank John Heasley, Job Snijders, Jared Mauch, Jakob Heitz, Keyur Patel, Douglas Montgomery, Randy Bush, Susan Hares, John Scudder, Curtis Villamizar, Danny McPherson, Chris Morrow, Tom Petch, Ilya Varlashkin, Enke Chen, Tony Li, Florian Weimer, John Leslie, Paul Jakma, Rob Austein, Russ Housley, Sandra Murphy, Steve Bellovin, Steve Kent, Steve Padgett, Alfred Hoenes, and Alvaro Retana for comments and suggestions.

9. References

9.1. Normative References

- [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>>.

- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC5065] Traina, P., McPherson, D., and J. Scudder, "Autonomous System Confederations for BGP", RFC 5065, DOI 10.17487/RFC5065, August 2007, <<https://www.rfc-editor.org/info/rfc5065>>.

9.2. Informative References

- [Analysis] Hannachi, L. and K. Sriram, "Detailed analysis of AS_SETs in BGP updates", NIST Robust Inter-domain Routing Project Website , October 2019, <https://github.com/ksriram25/IETF/blob/main/Detailed-AS_SET-analysis.txt>.
- [IANA-SP-ASN] "Special-Purpose Autonomous System (AS) Numbers", <<https://www.iana.org/assignments/iana-as-numbers-special-registry/iana-as-numbers-special-registry.xhtml>>.
- [RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, DOI 10.17487/RFC3779, June 2004, <<https://www.rfc-editor.org/info/rfc3779>>.
- [RFC6472] Kumari, W. and K. Sriram, "Recommendation for Not Using AS_SET and AS_CONFED_SET in BGP", BCP 172, RFC 6472, DOI 10.17487/RFC6472, December 2011, <<https://www.rfc-editor.org/info/rfc6472>>.
- [RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <<https://www.rfc-editor.org/info/rfc6480>>.
- [RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", RFC 6811, DOI 10.17487/RFC6811, January 2013, <<https://www.rfc-editor.org/info/rfc6811>>.
- [RFC7606] Chen, E., Ed., Scudder, J., Ed., Mohapatra, P., and K. Patel, "Revised Error Handling for BGP UPDATE Messages", RFC 7606, DOI 10.17487/RFC7606, August 2015, <<https://www.rfc-editor.org/info/rfc7606>>.

- [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>>.
- [RFC8205] Lepinski, M., Ed. and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <<https://www.rfc-editor.org/info/rfc8205>>.

Authors' Addresses

Warren Kumari
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
United States of America
Phone: +1 571 748 4373
Email: warren@kumari.net

Kotikalapudi Sriram
USA NIST
100 Bureau Drive
Gaithersburg, MD 20899
United States of America
Phone: +1 301 975 3973
Email: ksriram@nist.gov

Lilia Hannachi
USA NIST
100 Bureau Drive
Gaithersburg, MD 20899
United States of America
Phone: +1 301 975 3259
Email: lilia.hannachi@nist.gov

Jeffrey Haas
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089
United States of America
Email: jhaas@juniper.net

INTERNET-DRAFT
Intended Status: Proposed Standard

W. Hao
Huawei Technologies
D. Eastlake
Futurewei Technologies
S. Litkowski
Cisco Systems
S. Zhuang
Huawei Technologies
April 18, 2022

Expires: October 17, 2022

BGP Dissemination of L2 Flow Specification Rules
draft-ietf-idr-flowspec-l2vpn-19

Abstract

This document defines a Border Gateway Protocol (BGP) Flow Specification (flowspec) extension to disseminate Ethernet Layer 2 (L2) and Layer 2 Virtual Private Network (L2VPN) traffic filtering rules either by themselves or in conjunction with L3 flowspecs. AFI/SAFI 6/133 and 25/134 are used for these purposes. New component types and two extended communities are also defined.

Status of This Document

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

Distribution of this document is unlimited. Comments should be sent to the authors or the IDR Working Group mailing list <idr@ietf.org>.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <https://www.ietf.org/lid-abstracts.html>. The list of Internet-Draft Shadow Directories can be accessed at <https://www.ietf.org/shadow.html>.

Table of Contents

1. Introduction.....	3
1.1 Terminology.....	4
2. Layer 2 Flow Specification Encoding.....	5
2.1 L2 Component Types.....	6
2.1.1 Type 1 - Ethernet Type (EtherType).....	6
2.1.2 Type 2 - Source MAC.....	7
2.1.3 Type 3 - Destination MAC.....	7
2.1.4 Type 4 - DSAP (Destination Service Access Point).....	7
2.1.5 Type 5 - SSAP (Source Service Access Point).....	7
2.1.6 Type 6 - Control field in LLC.....	8
2.1.7 Type 7 - SNAP.....	8
2.1.8 Type 8 - VLAN ID.....	8
2.1.9 Type 9 - VLAN PCP.....	8
2.1.10 Type 10 - Inner VLAN ID.....	9
2.1.11 Type 11 - Inner VLAN PCP.....	9
2.1.12 Type 12 - VLAN DEI.....	9
2.1.13 Type 13 - Inner VLAN DEI.....	10
2.1.14 Type 14 - Source MAC Special Bits.....	10
2.1.15 Type 15 - Destination MAC Special Bits.....	10
2.2 Order of Traffic Filtering Rules.....	10
3. L2VPN Flow Specification Encoding in BGP.....	12
3.1 Order of L2VPN Filtering Rules.....	12
4. Ethernet Flow Specification Traffic Actions.....	13
4.1 VLAN-action.....	13
4.2 TPID-action.....	15
5. Flow Spec Validation.....	16
6. IANA Considerations.....	17
7. Security Considerations.....	19
8. Acknowledgements.....	19
9. Contributors.....	19
Normative References.....	20
Informative References.....	21
Authors' Addresses.....	22

1. Introduction

Border Gateway Protocol (BGP) Flow Specification [RFC8955] (flowspec) is an extension to BGP that supports the dissemination of traffic flow specifications and resulting actions to be taken on packets in a specified flow. It leverages the BGP Control Plane to simplify the distribution of ACLs (Access Control Lists). Using the Flow Specification extension new filter rules can be injected to all BGP peers simultaneously without changing router configuration. A typical application is to automate the distribution of traffic filter lists to routers for DDoS (Distributed Denial of Service) mitigation, access control, and similar applications.

BGP Flow Specification [RFC8955] defines a BGP Network Layer Reachability Information (NLRI) format used to distribute traffic flow specification rules. The NLRI for (AFI=1, SAFI=133) specifies IPv4 unicast filtering. The NLRI for (AFI=1, SAFI=134) specifies IPv4 BGP/MPLS VPN filtering [RFC7432]. The Flow Specification match part defined in [RFC8955] only includes L3/L4 information like IPv4 source/destination prefix, protocol, ports, and the like, so traffic flows can only be filtered based on L3/L4 information. This has been extended by [RFC8956] to cover IPv6 (AFI=2) L3/L4.

Layer 2 Virtual Private Networks (L2VPNs) have been deployed in an increasing number of networks. Such networks also have requirements to deploy BGP Flow Specification to mitigate DDoS attack traffic. Within an L2VPN network, both IP and non-IP Ethernet traffic may exist. For IP traffic filtering, the VPN Flow Specification rules defined in [RFC8955] and/or [RFC8956], which include match criteria and actions, can still be used. For non-IP Ethernet traffic filtering, Layer 2 related information like source/destination MAC and VLAN must be considered.

There are different kinds of L2VPN networks like EVPN [RFC7432], BGP VPLS [RFC4761], LDP VPLS [RFC4762] and border gateway protocol (BGP) auto discovery [RFC6074]. Because the Flow Specification feature relies on the BGP protocol to distribute traffic filtering rules, it can only be incrementally deployed in those L2VPN networks where BGP has already been used for auto discovery and/or signaling purposes such as BGP-based VPLS [RFC4761], EVPN, and LDP-based VPLS [RFC4762] with BGP auto-discovery [RFC6074].

This document defines new flowspec component types and two new extended communities to support L2 and L2VPN flowspec applications. The flowspec rules can be enforced on all border routers or on some interface sets of the border routers. SAFI=133 in [RFC8955] and [RFC8956] is extended for AFI=6 as specified in Section 2 to cover L2 traffic filtering information and in Section 3 SAFI=134 is extended for AFI=25 to cover the L2VPN environment.

1.1 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 following acronyms and terms are used in this document:

AFI - Address Family Identifier

ACL - Access Control List

DDoS - Distributed Denial of Service

DEI - Drop Eligible Indicator

EVPN - Ethernet VPN [RFC7432]

flowspec - BGP Flow Specification

L2 - Layer 2

L2VPN - Layer 2 VPN

L3 - Layer 3

L3VPN - Layer 3 VPN

NLRI - Network Layer Reachability Information

PCP - Priority Code Point [802.1Q]

SAFI - Subsequent Address Family Identifier

TPID - Tag Protocol ID, typically a VLAN ID

VLAN - Virtual Local Area Network

VPLS - Virtual Private Line Service [RFC4762]

VPN - Virtual Private Network

2. Layer 2 Flow Specification Encoding

[RFC8955] defines SAFI 133 and SAFI 134, with AFI=1, for "dissemination of IPv4 flow specification rules" and "dissemination of VPNv4 flow specification rules", respectively. [RFC8956] extends [RFC8955] to also allow AFI=2 thus making it applicable to both IPv4 and IPv6 applications. This document further extends SAFI=133 for AFI=6 and SAFI=134 for AFI=25 to make them applicable to L2 and L2VPN applications. This document also provides for the optional combination of L3 flow specifications with these L2 flow specifications.

This section specifies the L2 flowspec for AFI=6/SAFI=133. To simplify assignments, a new registry is used for L2 flowspec. Since it is frequently desirable to also filter on L3/L4 fields, provision is made for their inclusion along with an indication of the L3 protocol involved (IPv4 or IPv6).

The NLRI part of the MP_REACH_NLRI and MP_UNREACH_NLRI is encoded as a 1- or 2-octet total NLRI length field followed by several fields as described below.

total-length (0xnn or 0xfnnn)	2 or 3 octets
L3-AFI	2 octets
L2-length (0xnn or 0xfnnn)	2 or 3 octets
NLRI-value	variable

Figure 1: Flow Specification NLRI for L2

The fields show in Figure 1 are further specified below:

total-length: The length of the subsequent fields (L3 AFI, L2-length, and NLRI-value) encoded as provided in Section 4.1 of [RFC8955]. If this field is less than 4, which is the minimum valid value, then the NLRI is malformed in which case a NOTIFICATION message is sent and the BGP connection closed as provided in Section 6.3 of [RFC4271].

L3-AFI: If no L3/L4 filtering is desired, this two octet field MUST be zero which is a reserved AFI value. Otherwise L3-AFI indicates the L3 protocol involved by giving its AFI (0x0001 for IPv4 or 0x0002 for IPv6). If the receiver does not understand the value of the L3-AFI field, the MP_REACH or MP_UNREACH attribute is ignored.

L2-length: The length of the L2 components at the beginning of the NLRI-value field encoded as provided in Section 4.1 of [RFC8955]. If the value of this field indicates that the L2 components extend beyond the total-length, the NLRI is malformed in which case a NOTIFICATION message is sent and the BGP connection closed as provided in Section 6.3 of [RFC4271]. N2-length MAY be zero although, in that case, it would have been more efficient to encode the attribute as an L3 Flow spec unless it is desired to apply an L2 action (see Section 4). A null L2 flowspec always matches.

NLRI-value: This consists of the L2 flowspec, of length L2-length, followed by an optionally present L3 flowspec. The result can be treated in most ways as a single flowspec, matching the intersection (AND) of all the components except that the components in the initial L2 region are interpreted as L2 components and the remainder as L3 components per the L3-AFI field. This is necessary because there are different registries for the L2, L3 IPv4, and L3 IPv6 component types. If the L3 flowspec is null (length zero), it always matches.

2.1 L2 Component Types

The L2 flowspec portion of the NLRI-value consists of flowspec components as in [RFC8955] but using L2 components and types as specified below. All components start with a type octet followed by a length octet followed by any additional information needed. The length octet gives the length, in octets, of the information after the length octet. This structure applies to all new components to be defined in the L2 Flow-spec Component Registry (see Section 6) and to all existing components except Types 2 and 3 where the length is in bits.

2.1.1 Type 1 - Ethernet Type (EtherType)

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match the two-octet EtherType field. op is encoded as specified in Section 4.2.1.1 of [RFC8955]. Values are encoded as 2-octet quantities. Ethernet II framing defines the two-octet Ethernet Type (EtherType) field in an Ethernet frame, preceded by destination and source MAC addresses, that identifies an upper layer protocol encapsulating the frame data. The match fails if LLC encoding is being used rather than EtherType encoding.

2.1.2 Type 2 - Source MAC

Encoding: <type (1 octet), MAC Prefix length (1 octet), MAC Prefix>

Defines the source MAC Address prefix to match encoded as in BGP UPDATE messages [RFC4271]. Prefix length is in bits and the MAC Prefix is fill out with from 1 to 7 padding bits so that it is an integer number of octets. These padding bits are ignored for matching purposes.

2.1.3 Type 3 - Destination MAC

Encoding: <type (1 octet), MAC Prefix length (1 octet), MAC Prefix>

Defines the destination MAC Address to match encoded as in BGP UPDATE messages [RFC4271]. Prefix length is in bits and the MAC Prefix is fill out with from 1 to 7 padding bits so that it is an integer number of octets. These padding bits are ignored for matching purposes.

2.1.4 Type 4 - DSAP (Destination Service Access Point)

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match the 1-octet DSAP in the IEEE 802.2 LLC (Logical Link Control Header). Values are encoded as 1-octet quantities. op is encoded as specified in Section 4.2.1.1 of [RFC8955]. The match fails if EtherType L2 header encoding is being used rather than LLC encoding.

2.1.5 Type 5 - SSAP (Source Service Access Point)

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match the 1-octet SSAP in the IEEE 802.2 LLC. Values are encoded as 1-octet quantities. op is encoded as specified in Section 4.2.1.1 of [RFC8955]. The match fails if EtherType L2 header encoding is being used rather than LLC encoding.

2.1.6 Type 6 - Control field in LLC

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match the 1-octet control field in the IEEE 802.2 LLC. Values are encoded as 1-octet quantities. op is encoded as specified in Section 4.2.1.1 of [RFC8955]. The match fails if EtherType L2 header encoding is being used rather than LLC encoding.

2.1.7 Type 7 - SNAP

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match 5-octet SNAP (Sub-Network Access Protocol) field. Values are encoded as 8-octet quantities with the zero padded SNAP left justified. op is encoded as specified in Section 4.2.1.1 of [RFC8955]. The match fails if EtherType L2 header encoding is being used rather than LLC encoding.

2.1.8 Type 8 - VLAN ID

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match VLAN ID. Values are encoded as 2-octet quantities, where the four most significant bits are set to zero and ignored for matching and the 12 least significant bits contain the VLAN value. op is encoded as specified in Section 4.2.1.1 of [RFC8955].

In the virtual local-area network (VLAN) stacking case, the VLAN ID is the outer VLAN ID.

2.1.9 Type 9 - VLAN PCP

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match 3-bit VLAN PCP (priority code point) fields [802.1Q]. Values are encoded using a single octet, where the five most significant bits are set to zero and ignored for matching and the three least significant bits contain the VLAN PCP value. op is encoded as specified in Section 4.2.1.1 of [RFC8955].

In the virtual local-area network (VLAN) stacking case, the VLAN PCP is part of the outer VLAN tag.

2.1.10 Type 10 - Inner VLAN ID

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match the inner VLAN ID for virtual local-area network (VLAN) stacking or Q-in-Q use. Values are encoded as 2-octet quantities, where the four most significant bits are set to zero and ignored for matching and the 12 least significant bits contain the VLAN value. op is encoded as specified in Section 4.2.1.1 of [RFC8955].

In the single VLAN case, this component type MUST NOT be used. If it appears the match will fail.

2.1.11 Type 11 - Inner VLAN PCP

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match 3-bit inner VLAN PCP fields [802.1Q] for virtual local-area network (VLAN) stacking or Q-in-Q use. Values are encoded using a single octet, where the five most significant bits are set to zero and ignored for matching and the three least significant bits contain the VLAN PCP value. op is encoded as specified in Section 4.2.1.1 of [RFC8955].

In the single VLAN case, this component type MUST NOT be used. If it appears the match will fail.

2.1.12 Type 12 - VLAN DEI

Encoding: <type (1 octet), length (1 octet), op (1 octet)>

This type tests the DEI (Drop Eligible Indicator) bit in the VLAN tag. If op is zero, it matches if and only if the DEI bit is zero. If op is non-zero, it matches if and only if the DEI bit is one.

In the virtual local-area network (VLAN) stacking case, the VLAN DEI is part of the outer VLAN tag.

2.1.13 Type 13 - Inner VLAN DEI

Encoding: <type (1 octet), length (1 octet), op (1 octet)>

This type tests the DEI bit in the inner VLAN tag. If op is zero, it matches if and only if the DEI bit is zero. If op is non-zero, it matches if and only if the DEI bit is one.

In the single VLAN case, this component type MUST NOT be used. If it appears the match will fail.

2.1.14 Type 14 - Source MAC Special Bits

Encoding: <type (1 octet), length (1 octet), op (1 octet)>

This type tests the bottom nibble of the top octet of the Source MAC address. The two low order bits of that nibble have long been the local bit (0x2) and the group addressed bit (0x1). However, recent changes in IEEE 802 have divided the local address space into 4 quadrants specified by the next two bits (0x4 and 0x8) [RFC7042bis]. This flowspec component permits testing, for example, that a MAC is group addressed or is a local address in a particular quadrant. The encoding is as given in Section 4.2.1.2 of [RFC8955].

2.1.15 Type 15 - Destination MAC Special Bits

Encoding: <type (1 octet), length (1 octet), op (1 octet)>

As discussed in Section 2.1.14 but for the Destination MAC Address special bits.

2.2 Order of Traffic Filtering Rules

The existing rules in Section 5.1 of [RFC8955] and in [RFC8956] for the ordering of traffic filtering are extended as follows:

L2 flowspecs (AFI = 6, 25) take precedence over L3 flowspecs (AFI = 1, 2). Between two L2 flowspecs, precedence of the L2 portion is determined as specified in this section after this paragraph. If the L2 flowspec L2 portions are the same and the L3-AFI is nonzero, then the L3 portions are compared as specified in [RFC8955] or [RFC8956] as appropriate. Note: if the L3-AFI fields are different between two L2 flowspecs, they will never match the same packet so it will not be necessary to prioritize two flowspecs with different L3-AFI values.

The original definition for the order of traffic filtering rules can be reused for L2 with new consideration for the MAC Address offset. As long as the offsets are equal, the comparison is the same, retaining longest-prefix-match semantics. If the offsets are not equal, the lowest offset has precedence, as this flow matches the most significant bit.

Pseudocode:

```
flow_rule_L2_cmp (a, b)
{
    comp1 = next_component(a);
    comp2 = next_component(b);
    while (comp1 || comp2) {
        // component_type returns infinity on end-of-list
        if (component_type(comp1) < component_type(comp2)) {
            return A_HAS_PRECEDENCE;
        }
        if (component_type(comp1) > component_type(comp2)) {
            return B_HAS_PRECEDENCE;
        }

        if (component_type(comp1) == MAC_DESTINATION || MAC_SOURCE) {
            common = MIN(MAC Address length (comp1),
                        MAC Address length (comp2));
            cmp = MAC Address compare(comp1, comp2, common);
            // not equal, lowest value has precedence
            // equal, longest match has precedence
        } else {
            common =
                MIN(component_length(comp1), component_length(comp2));
            cmp = memcmp(data(comp1), data(comp2), common);
            // not equal, lowest value has precedence
            // equal, longest string has precedence
        }
    }
    return EQUAL;
}
```

3. L2VPN Flow Specification Encoding in BGP

The NLRI format for AFI=25/SAFI=134 (L2VPN), as with the other VPN flowspec AFI/SAFI pairs, is the same as the non-VPN Flow-Spec but with the addition of a Route Distinguisher to identify the VPN to which the flowspec is to be applied.

In addition, the IANA entry for SAFI 134 is slightly generalized as specified at the beginning of Section 6.

The L2VPN NLRI format is as follows:

total-length (0xnn or 0xfnnn)	2 or 3 octets
Route Distinguisher	8 octets
L3-AFI	2 octets
L2-length (0xnn or 0xfnnn)	2 or 3 octets
NLRI-value	variable

Figure 2: Flow Specification NLRI for L2VPN

The fields in Figure 2, other than the Route Distinguisher, are encoded as specified in Section 2 except that the minimum value for total-length is 12.

Flow specification rules received via this NLRI apply only to traffic that belongs to the VPN instance(s) into which it is imported. Flow rules are accepted as specified in Section 5.

3.1 Order of L2VPN Filtering Rules

The order between L2VPN filtering rules is determined as specified in Section 2.2. Note that if the Route Distinguisher is different between two L2VPN filtering rules, they will never both match the same packet so they need not be prioritized.

4. Ethernet Flow Specification Traffic Actions

The default action for an L2 traffic filtering flowspec is to accept traffic that matches that particular rule. The following extended community values per [RFC8955] can be used to specify particular actions in an L2 VPN network:

type	extended community	encoding
0x8006	traffic-rate	2-octet as#, 4-octet float
0x8007	traffic-action	bitmask
0x8008	redirect	6-octet Route Target
0x8009	traffic-marking	DSCP value

Redirect: The action should be redefined to allow the traffic to be redirected to a MAC or IP VRF routing instance that lists the specified route-target in its import policy.

Besides the above extended communities, this document also specifies the following BGP extended communities for Ethernet flows to extend [RFC8955]:

type	extended community	encoding
TBD1	VLAN-action	bitmask
TBD2	TPID-action	bitmask

4.1 VLAN-action

The VLAN-action extended community, as shown in the diagram below, consists of 6 octets that include action Flags, two VLAN IDs, and the associated PCP and DEI values. The action Flags fields are further divided into two parts which correspond to the first action and the second action respectively. Bit 0 to bit 7 give the first action while bit 8 to bit 15 give the second action. The bits of PO, PU, SW, RI and RO in each part represent the action of Pop, Push, Swap, Rewrite inner VLAN and Rewrite outer VLAN respectively. Through this method, more complicated actions also can be represented in a single VLAN-action extended community, such as SwapPop, PushSwap, etc. For example, SwapPop action is the sequence of two actions, the first action is Swap and the second action is Pop.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PO1	PU1	SW1	RI1	RO1	Resv			PO2	PU2	SW2	RI2	RO2	Resv		
VLAN ID1												PCP1		DE1	
VLAN ID2												PCP2		DE2	

PO1: Pop action. If the PO1 flag is one, it indicates the outmost VLAN should be removed.

PU1: Push action. If PU1 is one, it indicates VLAN ID1 will be added, the associated PCP and DEI are PCP1 and DE1.

SW1: Swap action. If the SW1 flag is one, it indicates the outer VLAN and inner VLAN should be swapped.

PO2: Pop action. If the PO2 flag is one, it indicates the outmost VLAN should be removed.

PU2: Push action. If PU2 is one, it indicates VLAN ID2 will be added, the associated PCP and DEI are PCP2 and DE2.

SW2: Swap action. If the SW2 flag is one, it indicates the outer VLAN and inner VLAN should be swapped.

RI1 and RI2: Rewrite inner VLAN action. If the RIX flag is one (where "x" is "1" or "2"), it indicates the inner VLAN should be replaced by a new VLAN where the new VLAN is VLAN IDx and the associated PCP and DEI are PCPx and DEx. If the VLAN IDx is 0, the action is to only modify the PCP and DEI value of the inner VLAN.

RO1 and RO2: Rewrite outer VLAN action. If the ROx flag is one (where "x" is "1" or "2"), it indicates the outer VLAN should be replaced by a new VLAN where the new VLAN is VLAN IDx and the associated PCP and DEI are PCPx and DEx. If the VLAN IDx is 0, the action is to only modify the PCP and DEI value of the outer VLAN.

Resv: Reserved for future use. MUST be sent as zero and ignored on receipt.

Giving an example below: if the action of PUSH Inner VLAN 10 with PCP value 5 and DEI value 0 and PUSH Outer VLAN 20 with PCP value 6 and DEI value 0 is needed, the format of the VLAN-action extended community is as follows:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
10												1	0	1	0
20												1	1	0	0

4.2 TPID-action

The TPID-action extended community consists of 6 octets which includes the fields of action Flags, TP ID1 and TP ID2.

0	15
TI TO	Resv
	TP ID1
	TP ID2

TI: Mapping inner TP ID action. If the TI flag is one, it indicates the inner TP ID should be replaced by a new TP ID, the new TP ID is TP ID1.

TO: Mapping outer TP ID action. If the TO flag is one, it indicates the outer TP ID should be replaced by a new TP ID, the new TP ID is TP ID2.

Resv: Reserved for future use. MUST be sent as zero and ignored on receipt.

5. Flow Spec Validation

Flow Specifications received over AFI=25/SAFI=134 are validated against routing reachability received over AFI=25/SAFI=128 as modified to conform to [RFC9117].

6. IANA Considerations

IANA is requested to change the description for SAFI 134 [RFC8955] to read as follows and to change the reference for it to [this document]:

134 VPN dissemination of flow specification rules

IANA is requested to create an L2 Flow Specification Component Type registry on the Flow Spec Component Types registries web page as follows:

Name: L2 Flow Specification Component Types

Reference: [this document]

Registration Procedures:

0 Reserved
 1-127 Specification Required
 128-255 First Come First Served

Initial contents:

type	Reference	description
0	[this document]	Reserved
1	[this document]	Ethernet Type
2	[this document]	Source MAC
3	[this document]	Destination MAC
4	[this document]	DSAP in LLC
5	[this document]	SSAP in LLC
6	[this document]	Control field in LLC
7	[this document]	SNAP
8	[this document]	VLAN ID
9	[this document]	VLAN PCP
10	[this document]	Inner VLAN ID
11	[this document]	Inner VLAN PCP
12	[this document]	VLAN DEI
13	[this document]	Inner VLAN DEI
14	[this document]	Source MAC Special Bits
15	[this document]	Destination MAC Special Bits
16-254	[this document]	unassigned
255	[this document]	Reserved

IANA is requested to assign two values from the "BGP Extended Communities Type - extended, transitive" registry [suggested value provided in square brackets]:

Type value	Name	Reference
TBD1[0x080A]	Flow spec VLAN action	[this document]
TBD2[0x080B]	Flow spec TPID action	[this document]

7. Security Considerations

For General BGP Flow Specification Security Considerations, see [RFC8955].

VLAN tagging identifies Layer 2 communities which are commonly expected to be isolated except when higher layer connection is provided, such as Layer 3 routing. Thus, the ability of the flowspec VLAN action to change the VLAN ID in a frame might compromise security.

8. Acknowledgements

The authors wish to acknowledge the important contributions and suggestions of the following:

Hannes Gredler, Xiaohu Xu, Zhenbin Li, Lucy Yong, and Feng Dong.

9. Contributors

Qiandeng Liang
Huawei Technologies
101 Software Avenue, Yuhuatai District
Nanjing 210012
China

Email: liangqiandeng@huawei.com

Normative References

- [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>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <<https://www.rfc-editor.org/info/rfc4761>>.
- [RFC4762] Lasserre, M., Ed. and V. Kompella, Ed., "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, DOI 10.17487/RFC4762, January 2007, <<https://www.rfc-editor.org/info/rfc4762>>.
- [RFC6074] Rosen, E., Davie, B., Radoaca, V., and W. Luo, "Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)", RFC 6074, DOI 10.17487/RFC6074, January 2011, <<https://www.rfc-editor.org/info/rfc6074>>.
- [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>>.
- [RFC8955] Loibl, C., Hares, S., Raszuk, R., McPherson, D., and M. Bacher, "Dissemination of Flow Specification Rules", RFC 8955, DOI 10.17487/RFC8955, December 2020, <<https://www.rfc-editor.org/info/rfc8955>>.
- [RFC8956] Loibl, C., Ed., Raszuk, R., Ed., and S. Hares, Ed., "Dissemination of Flow Specification Rules for IPv6", RFC 8956, DOI 10.17487/RFC8956, December 2020, <<https://www.rfc-editor.org/info/rfc8956>>.
- [RFC9117] Uttaro, J., Alcaide, J., Filsfils, C., Smith, D., and P. Mohapatra, "Revised Validation Procedure for BGP Flow Specifications", RFC 9117, DOI 10.17487/RFC9117, August 2021, <<https://www.rfc-editor.org/info/rfc9117>>.

Informative References

- [RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <<https://www.rfc-editor.org/info/rfc7432>>.
- [RFC7042bis] Eastlake, D., and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", draft-eastlake-rfc7042bis, work in progress, March 2021.

Authors' Addresses

Weiguo Hao
Huawei Technologies
101 Software Avenue,
Nanjing 210012
China

Email: haoweiguo@huawei.com

Donald E. Eastlake, 3rd
Futurewei Technologies
2386 Panoramic Circle
Apopka, FL 32703
USA

Tel: +1-508-333-2270
Email: d3e3e3@gmail.com

Stephane Litkowski
Cisco Systems, Inc.

Email: slitkows.ietf@gmail.com

Shunwan Zhuang
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: zhuangshunwan@huawei.com

Copyright, Disclaimer, and Additional IPR Provisions

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

INTERNET-DRAFT
Intended Status: Proposed Standard

D. Eastlake
Futurewei Technologies
W. Hao
S. Zhuang
Z. Li
Huawei Technologies
R. Gu
China Mobile
February 6, 2022

Expires: August 5, 2022

BGP Dissemination of
Flow Specification Rules for Tunneled Traffic
draft-ietf-idr-flowspec-nvo3-15

Abstract

This draft specifies a Border Gateway Protocol (BGP) Network Layer Reachability Information (NLRI) encoding format for flow specifications (RFC 8955) that can match on a variety of tunneled traffic. In addition, flow specification components are specified for certain tunneling header fields.

Status of This Document

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

Distribution of this document is unlimited. Comments should be sent to the authors or the IDR Working Group mailing list <idr@ietf.org>.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <https://www.ietf.org/lid-abstracts.html>. The list of Internet-Draft Shadow Directories can be accessed at <https://www.ietf.org/shadow.html>.

Table of Contents

1. Introduction.....	3
1.1 Terminology.....	3
2. Tunneled Traffic Flow Specification NLRI.....	5
2.1 The SAFI Code Point.....	7
2.2 Tunnel Header Component Code Points.....	7
2.3 Specific Tunnel Types.....	9
2.3.1 VXLAN.....	9
2.3.2 VXLAN-GPE.....	10
2.3.3 NVGRE.....	11
2.3.4 L2TPv3.....	11
2.3.4.1 L2TPv3 Data Messages.....	12
2.3.4.2 L2TPv3 Control Messages.....	12
2.3.5 GRE.....	12
2.3.6 IP-in-IP.....	13
2.3.7 Geneve.....	14
2.4 Tunneled Traffic Actions.....	14
3. Order of Traffic Filtering Rules.....	15
4. Flow Spec Validation.....	16
5. Security Considerations.....	16
6. IANA Considerations.....	17
Normative References.....	18
Informative References.....	19
Acknowledgments.....	20
Authors' Addresses.....	20

1. Introduction

BGP Flow Specification (flowspec [RFC8955]) is an extension to BGP that supports the dissemination of traffic flow specification rules. It uses the BGP control plane to simplify the distribution of Access Control Lists (ACLs) and allows new filter rules to be injected to all BGP peers simultaneously without changing router configuration. A typical application of BGP flowspec is to automate the distribution of traffic filter lists to routers for Distributed Denial of Service (DDOS) mitigation.

BGP flowspec defines BGP Network Layer Reachability Information (NLRI) formats used to distribute traffic flow specification rules. AFI=1/SAFI=133 is for IPv4 unicast filtering. AFI=1/SAFI=134 is for IPv4 BGP/MPLS VPN filtering [RFC8955]. [RFC8956] and [FlowSpecL2] extend the flowspec rules for IPv6 and Layer 2 Ethernet packets respectively. None of these previously defined flow specifications are suitable for matching in cases of tunneling or encapsulation where there might be duplicates of a layer of header such as two IPv6 headers in IP-in-IP [RFC2003] or a nested header sequence such as the Layer 2 and 3 headers encapsulated in VXLAN [RFC7348].

In the cloud computing era, multi-tenancy has become a core requirement for data centers. It is increasingly common to see tunneled traffic with a field to distinguish tenants. An example is the Network Virtualization Over Layer 3 (NVO3 [RFC8014]) overlay technology that can satisfy multi-tenancy key requirements. VXLAN [RFC7348] and NVGRE [RFC7637] are two typical NVO3 encapsulations. Other encapsulations such as IP-in-IP or GRE may be encountered. Because these tunnel / overlay technologies involving an additional level of encapsulation, flow specification that can match on the inner header as well as the outer header and fields in any tunneling header are needed.

In summary, Flow Specifications should be able to include inner nested header information as well as fields specific to the type of tunneling in use such as virtual network / tenant ID. This draft specifies methods for accomplishing this using SAFI=77 and a new NLRI encoding. In addition, flow specification components are specified for certain tunneling header fields.

1.1 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 reader is assumed to be familiar with BGP terminology [RFC4271] [RFC4760]. The following terms and acronyms are used in this document with the meaning indicated:

ACL - Access Control List

DDoS - Distributed Denial of Service (Attack)

DSCP - Differentiated Services Code Point [RFC2474]

GRE - Generic Router Encapsulation [RFC2890]

L2TPv3 - Layer Two Tunneling Protocol - Version 3 [RFC3931]

NLRI - Network Layer Reachability Information [RFC4271] [RFC4760]

NVGRE - Network Virtualization Using Generic Routing Encapsulation [RFC7637]

NVO3 - Network Virtual Overlay Layer 3 [RFC8014]

PE - Provider Edge

VN - virtual network

VXLAN - Virtual eXtensible Local Area Network [RFC7348]

2. Tunnelled Traffic Flow Specification NLRI

The Flowspec rules specified in [RFC8955], [RFC8956], and [FlowSpecL2] cannot match or filter tunneled traffic based on the tunnel type, any tunnel header fields, or headers past the tunnel header. To enable flow specification of tunneled traffic, a new SAFI (77) and NLRI encoding are specified. This encoding, shown in Figure 1, enables flow specification of more than one layer of header when needed.

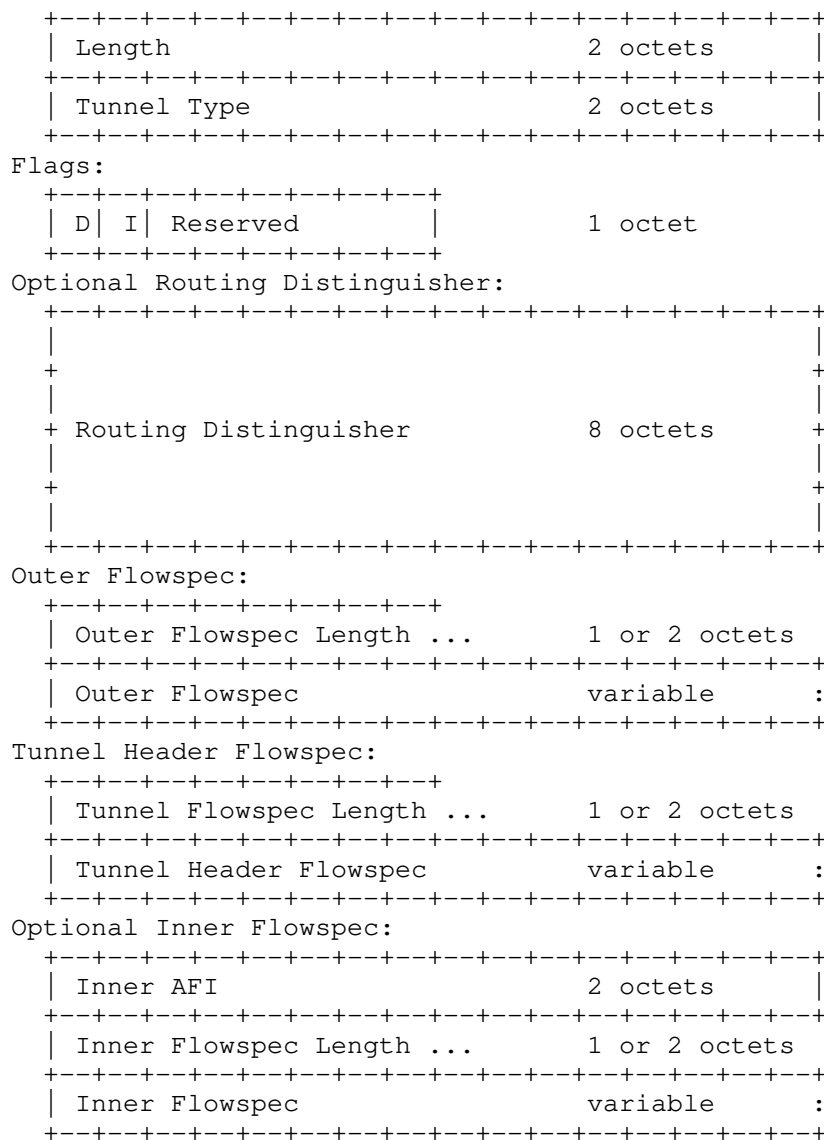


Figure 1. Tunneled Traffic Flowspec NLRI

- Length - The NLRI Length including the Tunnel Type encoded as an unsigned integer.
- Tunnel Type - The type of tunnel using a value from the IANA BGP Tunnel Encapsulation Attribute Tunnel Types registry.
- Flags: D bit - Indicates the presence of the Routing Distinguisher (see below).
- Flags: I bit - Indicates the presence of the Inner AFI and the Inner Flowspec (see below).
- Flags: Reserved - Six bits that MUST be sent as zero and ignored on receipt.
- Routing Distinguisher - If the outer Layer 3 address belongs to a BGP/MPLS VPN, the routing distinguisher is included to indicate traffic filtering within that VPN. Because NVO3 outer layer addresses normally belong to a public network, a Route Distinguisher field is normally not needed for NVO3.
- Outer Flowspec / Length - The flow specification for the outer header. The length is encoded as provided in Section 4.1 of [RFC8955]. The AFI for the Outer Flowspec is the AFI at the beginning of the BGP multiprotocol MP_REACH_NLRI or MP_UNREACH_NLRI containing the tunneled traffic flow specification NLRI.
- Tunnel Header Flowspec / Length - The flow specification for the tunneling header. The length is encoded as provided in Section 4.1 of [RFC8955]. This specifies matching criterion on tunnel header fields as well as, implicitly, on the tunnel type which is indicated by the Tunnel Type field above. For some types of tunneling, such as IP-in-IP, there may be no tunnel header fields. For other types of tunneling, there may be several tunnel header fields on which matching can be specified with this flowspec. If a Tunnel Type has no tunnel header fields or it is not desired to filter on header fields, the Tunnel Flowspec length field is present but has value zero.
- Inner AFI - Depending on the Tunnel Type, there may be an Inner AFI that indicate the type of inner flow specification. The "Inner SAFI" is implicitly 133 for flowspec.
- Inner Flowspec / Length - Depending on the Tunnel Type, there may be an inner flowspec for the header level encapsulated within the outer header. The length is encoded as provided in Section 4.1 of [RFC8955].

A Tunneled Traffic Flowspec matches if the Outer Flowspec, Tunnel Type, and Tunnel Header Flowspec match and, in addition, each of the following optional items that is present matches:

- Inner Flowspec, and
- Routing Distinguisher.

An omitted (as can be done for the Inner Flowspec) or null flowspec is considered to always match.

2.1 The SAFI Code Point

Use of the tunneled traffic flow specification NLRI format is indicated by SAFI=77. This is used in conjunction with the AFI for the outer header, that is AFI=1 for IPv4, AFI=2 for IPv6, and AFI=6 for Layer 2.

2.2 Tunnel Header Component Code Points

For most cases of tunneled traffic, there are tunnel header fields that can be tested by components that appear in the Tunnel Header Flowspec field. The types for these components are specified in a Tunnel Header Flowspec component registry (see Section 6) and the initial entries in this registry are specified below.

All Tunnel Header field components defined below and all such components added in the future have a TLV structure as follows:

- one octet of type followed by
- one octet giving the length of the value part as an unsigned integer number of octets followed by
- the specific matching operations/values as determined by the type.

Type 1 - VN ID

Encoding: <type (1 octet), length (1 octet), [op, value]+>.

Defines a list of {operation, value} pairs used to match the 24-bit VN ID that is used as the tenant identification in some tunneling headers. For VXLAN and Geneve encapsulation, the VN ID field is the VNI. For NVGRE encapsulation, the VN ID is the VSID. op is encoded as specified in Section 4.2.3 of [RFC8955]. Values are encoded as a 1, 2, or 4 octet quantity. If value is 24-bits, it is left-justified in the first 3 octets of the value and the last value octet MUST be sent as zero and ignored on receipt.

Type 2 - Flow ID

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match 8-bit Flow ID fields which are currently only useful for NVGRE encapsulation. op is encoded as specified in Section 4.2.3 of [RFC8955]. Values are encoded as a 1-octet quantity.

Type 3 - Session

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match a 32-bit Session field. This field is called Key in GRE [RFC2890] encapsulation and Session ID in L2TPv3 encapsulation. op is encoded as specified in Section 4.2.3 of [RFC8955]. Values are encoded as a 1, 2, or 4 octet quantity; if 1 or 2 octets are provided, these are right justified and padded on the left with zeros.

Type 4 - Cookie

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match a variable length Cookie field. This is only useful in L2TPv3 encapsulation. op is encoded as specified in Section 4.2.3 of [RFC8955]. Values are encoded as a 1, 2, 4, or 8 octet quantity. If the Cookie does not fit exactly into the value length, it is left justified and padded with following octets that MUST be sent as zero and ignored on receipt.

Type 5 - Tunnel Header Flags

Encoding: <type (1 octet), length (1 octet), [op, bitmask]+>

Defines a list of {operation, bitmask} pairs used to match against the tunnel header flags field. op is encoded as in Section 4.2.9 of [RFC8955]. bitmask is encoded as 1 octet for VXLAN-GPE and Geneve and as 2 octets for L2TPv3 control messages. When matching on L2TPv3 control message flags, the 3-bit Version subfield is treated as if it was zero.

Type 6 - L2TP Control Version

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match against the L2TP Control Message Version. op is encoded as in Section 4.2.3 of [RFC8955]. Value is encoded as 1 octet.

Type 7 - L2TPv3 Control Connection ID

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match

against the L2TPv3 Control Connection ID. op is encoded as in Section 4.2.3 of [RFC8955]. Value is encoded as 4 octets.

Type 8 - L2TPv3 Ns

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match against the L2TPv3 control message Ns field. op is encoded as in Section 4.2.3 of [RFC8955]. Value is encoded as 2 octets.

Type 9 - L2TPv3 Nr

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match against the L2TPv3 control message Nr field. op is encoded as in Section 4.2.3 of [RFC8955]. Values are encoded as 2 octets.

Type 10 - Protocol Type

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match against the GRE and Geneve Protocol Type fields. op is encoded as in Section 4.2.3 of [RFC8955]. Values are encoded as 2 octets.

Type 11 - GRE Sequence

Encoding: <type (1 octet), length (1 octet), [op, value]+>

Defines a list of {operation, value} pairs used to match against the GRE Sequence field. op is encoded as in Section 4.2.3 of [RFC8955]. Values are encoded as a 1, 2, or 4 octet quantity; if 1 or 2 octets are provided, these are right justified and padded on the left with zeros.

2.3 Specific Tunnel Types

The following subsections describe how to handle flow specification for several specific tunnel types.

2.3.1 VXLAN

The headers on a VXLAN [RFC7348] data packet are an outer Ethernet header, an outer IP header, a UDP header, the VXLAN header, and an inner Ethernet header. This inner Ethernet header is frequently, but not always, followed by an inner IP header. If the tunnel type is VXLAN, the I flag MUST be set in the Tunneled Traffic Flow

Specification.

If the outer Ethernet header is not being matched, the version (IPv4 or IPv6) of the outer IP header is indicated by the AFI at the beginning of the multiprotocol MP_REACH_NLRI or MP_UNREACH_NLRI containing the Tunneled Traffic Flow Specification NLRI. The outer Flowspec is used to filter the outer headers including, if desired, the UDP header.

If the outer Ethernet header is being matched, then the initial AFI is 6 [FlowSpecL2] and the Outer Flowspec can match the outer Ethernet header, specify the IP version of the outer IP header, and match that IP header including, if desired, the UDP header.

The Tunnel Header Flowspec can be used to filter on the VXLAN header VN ID (VNI).

The Inner Flowspec can be used on the Inner Ethernet header [FlowSpecL2] and any following IP header. If the inner AFI is 6, then the inner Flowspec provides filtering of the Layer 2 header, indicates whether filtering on a following IPv4 or IPv6 header is desired, and if it is desired provides the Flowspec components for that filtering. If the Inner AFI is 1 or 2, the Inner Ethernet header is not matched and to match the Flowspec the Inner Ethernet header must be followed by an IPv4 or IPv6 header, respectively, and the inner Flowspec is used to filter that inner IP header.

The inner MAC/IP address is associated with the VN ID. In the NVO3 terminating into a VPN scenario, if multiple access VN IDs map to one VPN instance, one shared VN ID can be carried in the flowspec rule to enforce the rule on the entire VPN instance and the shared VN ID and VPN correspondence should be configured on each VPN PE beforehand. In this case, the function of the Layer 3 VN ID is the same as a Route Distinguisher: it acts as the identification of the VPN instance.

2.3.2 VXLAN-GPE

VXLAN-GPE [GPE] is similar to VXLAN. The VXLAN-GPE header is the same size as the VXLAN header but has been extended from the VXLAN header by specifying a number of bits that are reserved in the VXLAN header. In particular, a number of additional flag bits are specified and a Next Protocol field is added that is valid if the P flag bit is set in the VXLAN-GPE header. These flags bits can be tested using the Tunnel Header Flags flowspec component defined above. VXLAN and VXLAN-GPE are distinguished by the port number in the UDP header the precedes the VXLAN or VXLAN-GPE headers.

If the VXLAN-GPE header P flag is zero, then that header is followed

by the same sequence as for VXLAN and the same flowspec choices apply (see Section 2.3.1).

If the VXLAN-GPE header P flag is one and that header's next protocol field is 1, then the VXLAN-GPE header is followed by an IPv4 header (there is no Inner Ethernet header). The Inner Flowspec matches only if the Inner AFI is 1 and the Inner Flowspec matches.

If the VXLAN-GPE header P flag is one and that header's next protocol field is 2, then the VXLAN-GPE header is followed by an IPv6 header (there is no Inner Ethernet header). The Inner Flowspec match only if the Inner AFI is 2 and the Inner Flowspec matches.

2.3.3 NVGRE

NVGRE [RFC7637] is similar to VXLAN except that the UDP header and VXLAN header immediately after the outer IP header are replaced by a GRE (Generic Router Encapsulation) header. The GRE header as used in NVGRE has no Checksum or Reserved1 field as shown in [RFC2890] but there are Virtual Subnet ID and Flow ID fields in place of what is labeled in [RFC2890] as the Key field. Processing and restrictions for NVGRE are as in Section 2.3.1 eliminating references to a UDP header and replacing references to the VXLAN header and its VN ID with references to the GRE header and its VN ID (VSID) and Flow ID.

2.3.4 L2TPv3

The headers on an L2TPv3 [RFC3931] packets are an outer Ethernet header, an outer IP header, the L2TPv3 header, an inner Ethernet header, and possibly an inner IP header if indicated by the inner Ethernet header EtherType. The Outer Flowspec operates on the outer headers that precede the L2TPv3 Session Header. The version of IP in the outer IP header is specified by either the outer AFI at the beginning of the MP_REACH_NLRI or MP_UNREACH_NLRI or, if that AFI is 6 (L2), optionally specified by the inner AFI within that L2 flowspec.

L2TPv3 data messages and control messages both start with a Session ID and are distinguished by whether the Session ID is non-zero or zero, respectively. Data message filtering is further specified in Section 2.3.4.1 and control message filtering is further specified in Section 2.3.4.2.

2.3.4.1 L2TPv3 Data Messages

For data messages, the L2TPv3 Session Header consists of a 32-bit non-zero Session ID followed by a variable length Cookie (maximum length 8 octets). A Tunnel Header flowspec is assumed to apply to data messages unless the first component requires a zero Session ID.

The Session ID and Cookie can be filtered on by using the Session and Cookie flowspec components in the Tunnel Header Flowspec. To filter on Cookie or even be able to bypass Cookie and parse the remainder of the L2TPv3 packet, the node implementing tunneled traffic flowspec needs to know the length and/or value of the Cookie fields of interest. This is negotiated at L2TPv3 session establishment and it is out of scope for this document how the node would learn this information. Of course, if flowspec is being used for DDOS mitigation and the Cookie has a fixed length and/or value in the DDOS traffic, this could be learned by inspecting that traffic.

If the I flag bit is zero, then no filtering is done on data beyond the L2TPv3 header. If the I flag is one, indicating the presence of an Inner Flowspec, and the node implementing flowspec does not know the length of the L2TPv3 header Cookie, the match fails. If that node does know the length of that Cookie, the Inner Flowspec is matched against the headers at the beginning of that data using the Inner AFI. If that Inner AFI is 1 or 2, then an inner IP header is required and filtering can be done on that IPv4 or IPv6 header respectively. If the Inner AFI is 6, filtering is done on the inner Ethernet header and, if an IPv4 or IPv6 inner AFI is specified within the inner L2 flowspec, done on the following IP header [FlowSpecL2].

2.3.4.2 L2TPv3 Control Messages

Control messages are distinguished by starting with a zero value 32-bit Session ID. L2TPv3 control message flowspecs MUST start with a Session component that requires Session to be zero. For L2TPv3 control messages, there is no Cookie but there are L2TPv3 flags, a 3-bit Version field, a 32-bit Control Connection ID, and 16-bit Ns and Nr sequence numbers. These can be tested using the Tunnel Header Flags, L2TP Control Version, L2TPv3 Control Connection ID, L2TPv3 Ns, and L2TPv3 Nr flowspec components in the Tunnel Header Flowspec.

2.3.5 GRE

Generic Router Encapsulation (GRE [RFC2890]) is another type of encapsulation. The Outer Flowspec operates on the outer headers that precede the GRE header. The version of IP is specified by the outer

AFI at the beginning of the MP_REACH_NLRI or MP_UNREACH_NLRI.

The Tunnel Header Flags component can be used to match the first two octets of the GRE header. The Protocol Type component can be used to match the corresponding GRE header field. The Session and GRE Sequence components can be used to match on the GRE Key and GRE Sequence fields if those fields are present respectively. If either of those fields is not present, a component to match on that field fails.

If the I flag bit is zero, no filtering is done on data after the GRE header. If the I flag bit is one in the tunnel flowspec, then there is an inner AFI and inner flowspec and the Protocol Type field of the GRE header must correspond to the Inner AFI as follows for the tunnel Flowspec to match. Otherwise, the match fails.

GRE Protocol Type	Inner AFI
-----	-----
0x0800 (IPv4)	1
0x86DD (IPv6)	2
0x6558	6

With the I flag a one and the Inner AFI and GRE Protocol Type fields correspond, the Inner Flowspec is used to filter the inner IP headers (Inner AFI=1 or 2) or the inner Ethernet header and optionally a following IP header (Inner AFI=6).

2.3.6 IP-in-IP

IP-in-IP encapsulation [RFC2003] is indicated when an outer IP header indicates an inner IP IPv4 or IPv6 header by the value of the outer IP header's Protocol (IPv4) or Next Protocol (IPv6) field.

The IP version of the outer IP header (IPv4 or IPv6) matched is indicated by an AFI of 1 or 2 at the beginning of the MP_REACH_NLRI or MP_UNREACH_NLRI while if that AFI is 6, it indicates a match on the out Ethernet header and, optionally, the following IP Header [FlowSpecL2]. The IP version of the inner IP header is indicated by the Inner AFI and the Inner Flowspec applies to the inner IP header.

There is no tunnel header so there are no fields that can be matched by the Tunnel Header Flowspec in the case of IP-in-IP.

2.3.7 Geneve

The headers on a Geneve [RFC8926] encapsulated packet are an outer Ethernet header, an outer IP header, a UDP header, the Geneve header, and subsequent headers depending on the Geneve header Protocol Type field.

If the outer Ethernet header is not being matched, the version (IPv4 or IPv6) of the outer IP header is indicated by the AFI at the beginning of the multiprotocol MP_REACH_NLRI or MP_UNREACH_NLRI containing the Tunneled Traffic Flow Specification NLRI. The outer Flowspec is used to filter the outer headers including, if desired, the UDP header.

If the outer Ethernet header is being matched, then the initial AFI is 6 [FlowSpecL2] and the Outer Flowspec can match the outer Ethernet header, specify the IP version of the outer IP header, and match that IP header including, if desired, the UDP header.

The Tunnel Header Flowspec can be used to filter on the Protocol Type field and/or the VNI field in the Geneve header. The flags octet of the Geneve header, the second octet of that header, can be filtered using the Tunnel Header Flags component.

If an Inner Flowspec is present, it is used to match the header(s) after the Geneve header. The Protocol Type field in the Geneve header must correspond to the Inner AFI as shown in the table in Section 2.3.5 above or the match fails. If the Inner AFI and GRE Protocol Type fields correspond, the Inner Flowspec is used to filter the inner IP headers (Inner AFI=1 or 2) or the inner Ethernet header and optionally a following IP header (Inner AFI=6).

2.4 Tunneled Traffic Actions

The traffic filtering actions previously specified in [RFC8955] and [FlowSpecL2] are used for tunneled traffic. For Traffic Marking in NV03, only the DSCP in the outer header can be modified.

3. Order of Traffic Filtering Rules

The following rules determine which flowspec takes precedence where one or more are applicable and at least one of the applicable flowspecs is a tunneled traffic flowspec:

- In comparing an applicable tunneled traffic flow specification with an applicable non-tunneled flow specification, the tunneled specification has precedence.
- If comparing tunneled traffic flow specifications, if all are applicable, the tunnel types will be the same. Any that have a Routing Distinguisher will take precedence over those without a Routing Distinguisher. Of those with a Routing Distinguisher, all applicable flowspecs will have the same Routing Distinguisher.
- At this point in the process, all remaining contenders for the highest precedence will either not have a Routing Distinguisher or have equal Routing Distinguishers. If more than one contender remain, those with an L2 Outer Flowspec take precedence over those with an L3 Outer Flowspec. If the Outer Flowspec AFI is the same, their order of precedence is determined by comparing the Outer Flowspecs as described in [RFC8955] and [RFC8956] for AFI for 1 or 2 respectively or [FlowSpecL2] for AFI=6.
- If the Outer Flowspecs are equal, then the Tunnel Header Flowspecs are compared using the usual sequential component comparison process [RFC8955].
- If the Tunnel Header Flowspecs are equal then compare the "I" flag. Those with an Inner Flowspec take precedence over those without an Inner Flowspec. If you get to this stage in the ordering process, those without an Inner Flowspec are equal. For those with an Inner Flowspec, check the Inner AFI. An L2 Inner AFI (AFI=6) takes precedence over an L3 Inner AFI.
- If the Inner AFIs are equal, precedence is determined by comparing the Inner Flowspecs as described in [FlowSpecL2] for L2 or [RFC8955] for L3.

4. Flow Spec Validation

Flowspecs received over AFI=1/SAFI=77 or AFI=2/SAFI=77 are validated, using only the Outer Flowspec, against routing reachability received over AFI=1/SAFI=133 and AFI=2/SAFI=133 respectively, as modified by [RFC9117].

5. Security Considerations

No new security issues are introduced to the BGP protocol by this specification.

For general Flowspec security considerations, see [RFC8955].

6. IANA Considerations

IANA has assigned the following SAFI:

Value	Description	Reference
77	Tunneled Traffic Flowspec	[This document]

IANA is requested to create a Tunnel Header Flow Spec Component Type registry on the Flow Spec Component Types registries web page as follows:

Name: Tunnel Flow Spec Component Types

Reference: [this document]

Registration Procedures:

0	Reserved
1-127	Specification Required
128-254	First Come First Served
255	Reserved

Initial contents:

Type	Name	Reference
0	reserved	[this document]
1	VN ID	[this document]
2	Flow ID	[this document]
3	Session	[this document]
4	Cookie	[this document]
5	Tunnel Header Flags	[this document]
6	L2TP Control Version	[this document]
7	L2TPv3 Control Connection ID	[this document]
8	L2TPv3 Ns	[this document]
9	L2TPv3 Nr	[this document]
10	Protocol Type	[this document]
11	GRE Sequence	[this document]
12-254	unassigned	[this document]
255	reserved	[this document]

Normative References

- [RFC2003] - Perkins, C., "IP Encapsulation within IP", RFC 2003, DOI 10.17487/RFC2003, October 1996, <<https://www.rfc-editor.org/info/rfc2003>>.
- [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>>.
- [RFC2474] - Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <<https://www.rfc-editor.org/info/rfc2474>>.
- [RFC2890] - Dommety, G., "Key and Sequence Number Extensions to GRE", RFC 2890, DOI 10.17487/RFC2890, September 2000, <<https://www.rfc-editor.org/info/rfc2890>>.
- [RFC3931] - Lau, J., Ed., Townsley, M., Ed., and I. Goyret, Ed., "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, DOI 10.17487/RFC3931, March 2005, <<https://www.rfc-editor.org/info/rfc3931>>.
- [RFC4271] - Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC4760] - Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <<https://www.rfc-editor.org/info/rfc4760>>.
- [RFC7348] - Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <<https://www.rfc-editor.org/info/rfc7348>>.
- [RFC7637] - Garg, P., Ed., and Y. Wang, Ed., "NVGRE: Network Virtualization Using Generic Routing Encapsulation", RFC 7637, DOI 10.17487/RFC7637, September 2015, <<https://www.rfc-editor.org/info/rfc7637>>.
- [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>>.

- [RFC8926] - Gross, J., Ed., Ganga, I., Ed., and T. Sridhar, Ed., "Geneve: Generic Network Virtualization Encapsulation", RFC 8926, DOI 10.17487/RFC8926, November 2020, <<https://www.rfc-editor.org/info/rfc8926>>.
- [RFC8955] - Loibl, C., Hares, S., Raszuk, R., McPherson, D., and M. Bacher, "Dissemination of Flow Specification Rules", RFC 8955, DOI 10.17487/RFC8955, December 2020, <<https://www.rfc-editor.org/info/rfc8955>>.
- [RFC8956] - Loibl, C., Ed., Raszuk, R., Ed., and S. Hares, Ed., "Dissemination of Flow Specification Rules for IPv6", RFC 8956, DOI 10.17487/RFC8956, December 2020, <<https://www.rfc-editor.org/info/rfc8956>>.
- [RFC9117] - Uttaro, J., Alcaide, J., Filsfils, C., Smith, D., and P. Mohapatra, "Revised Validation Procedure for BGP Flow Specifications", RFC 9117, DOI 10.17487/RFC9117, August 2021, <<https://www.rfc-editor.org/info/rfc9117>>.
- [FlowSpecL2] - W. Hao, et al, "Dissemination of Flow Specification Rules for L2 VPN", draft-ietf-idr-flowspec-l2vpn, work in progress.

Informative References

- [RFC8014] - Black, D., Hudson, J., Kreeger, L., Lasserre, M., and T. Narten, "An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)", RFC 8014, DOI 10.17487/RFC8014, December 2016, <<https://www.rfc-editor.org/info/rfc8014>>.
- [GPE] - P. Quinn, et al, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe, work in progress.

Acknowledgments

The authors wish to acknowledge the important contributions of the following listed in alphabetic order:

Jeff Haas, Susan Hares, Yizhou Li, Qiandeng Liang, Greg Mirsky,
Nan Wu, Robert Raszuk, and Lucy Yong

Authors' Addresses

Donald Eastlake
Futurewei Technologies
2386 Panoramic Circle
Apopka, FL 32703 USA

Tel: +1-508-333-2270
Email: d3e3e3@gmail.com

Weiguo Hao
Huawei Technologies
101 Software Avenue,
Nanjing 210012 China

Email: haoweiguo@huawei.com

Shunwan Zhuang
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095 China

Email: zhuangshunwan@huawei.com

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095 China

Email: lizhenbin@huawei.com

Rong Gu
China Mobile

Email: gurong_cmcc@outlook.com

Copyright, Disclaimer, and Additional IPR Provisions

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Network Working Group
Internet-Draft
Updates: 9012 (if approved)
Intended status: Standards Track
Expires: October 16, 2022

S. Previdi
Huawei Technologies
C. Filsfils
Cisco Systems
K. Talaulikar, Ed.
Arrcus Inc
P. Mattes
Microsoft
D. Jain
S. Lin
Google
April 14, 2022

Advertising Segment Routing Policies in BGP
draft-ietf-idr-segment-routing-te-policy-17

Abstract

This document defines a new BGP SAFI with a new NLRI to advertise a candidate path of a Segment Routing (SR) Policy. An SR Policy is a set of candidate paths, each consisting of one or more segment lists. The headend of an SR Policy may learn multiple candidate paths for an SR Policy. Candidate paths may be learned via several different mechanisms, e.g., CLI, NetConf, PCEP, or BGP. This document specifies how BGP may be used to distribute SR Policy candidate paths. New sub-TLVs for the Tunnel Encapsulation Attribute are defined for signaling information about these candidate paths.

This documents updates RFC9012 with extensions to the Color Extended Community to support new steering modes over SR Policy.

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 October 16, 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction	3
1.1. Requirements Language	5
2. SR Policy Encoding	5
2.1. SR Policy SAFI and NLRI	5
2.2. SR Policy and Tunnel Encapsulation Attribute	7
2.3. Remote Endpoint and Color	8
2.4. SR Policy Sub-TLVs	9
2.4.1. Preference Sub-TLV	9
2.4.2. Binding SID Sub-TLV	10
2.4.3. SRv6 Binding SID Sub-TLV	11
2.4.4. Segment List Sub-TLV	13
2.4.5. Explicit NULL Label Policy Sub-TLV	27
2.4.6. Policy Priority Sub-TLV	29
2.4.7. Policy Candidate Path Name Sub-TLV	30
2.4.8. Policy Name Sub-TLV	31
3. Color Extended Community	32
4. SR Policy Operations	33
4.1. Advertisement of SR Policies	33
4.2. Reception of an SR Policy NLRI	33
4.2.1. Acceptance of an SR Policy NLRI	33
4.2.2. Usable SR Policy NLRI	34
4.2.3. Passing a usable SR Policy NLRI to the SRPM	34
4.2.4. Propagation of an SR Policy	35
5. Error Handling	35
6. IANA Considerations	36
6.1. Existing Registry: Subsequent Address Family Identifiers (SAFI) Parameters	37
6.2. Existing Registry: BGP Tunnel Encapsulation Attribute Tunnel Types	37
6.3. Existing Registry: BGP Tunnel Encapsulation Attribute sub-TLVs	37

6.4.	Existing Registry: Color Extended Community Flags	38
6.5.	New Registry: SR Policy Segment List Sub-TLVs	38
6.6.	New Registry: SR Policy Binding SID Flags	39
6.7.	New Registry: SR Policy SRv6 Binding SID Flags	39
6.8.	New Registry: SR Policy Segment Flags	40
6.9.	New Registry: Color Extended Community Color-Only Types .	40
7.	Security Considerations	41
8.	Acknowledgments	41
9.	Contributors	42
10.	References	43
10.1.	Normative References	43
10.2.	Informational References	44
	Authors' Addresses	45

1. Introduction

Segment Routing (SR) [RFC8402] allows a headend node to steer a packet flow along any path. Intermediate per-path states are eliminated thanks to source routing.

The headend node is said to steer a flow into an SR Policy [RFC8402].

The packets steered into an SR Policy carry an ordered list of segments associated with that SR Policy.

[I-D.ietf-spring-segment-routing-policy] details the concepts of SR Policy and steering into an SR Policy. These apply equally to the SR-MPLS and Segment Routing for IPv6 (SRv6) data-plane instantiations of Segment Routing using SR-MPLS and SRv6 Segment Identifiers (SIDs) as described in [RFC8402]. [RFC8660] describes the representation and processing of this ordered list of segments as MPLS label stack for SR-MPLS. While [RFC8754] and [RFC8986] describe the same for SRv6 with the use of the Segment Routing Header (SRH).

The SR Policy related functionality described in [I-D.ietf-spring-segment-routing-policy] can be conceptually viewed as being incorporated in an SR Policy Module (SRPM). Following is a reminder of the high-level functionality of SRPM:

- o Learning multiple candidate paths for an SR Policy via various mechanisms (CLI, NetConf, PCEP or BGP).
- o Selection of the best candidate path for an SR Policy.
- o Binding BSID to the selected candidate path of an SR Policy.
- o Installation of the selected candidate path and its BSID in the forwarding plane.

This document specifies the way to use BGP to distribute one or more of the candidate paths of an SR Policy to the headend of that policy. The document describes the functionality provided by BGP and, as appropriate, provides references for the functionality which is outside the scope of BGP (i.e. resides within SRPM on the headend node).

This document specifies a way of representing SR Policy candidate paths in BGP UPDATE messages. BGP can then be used to propagate the SR Policy candidate paths to the headend nodes in the network. The usual BGP rules for BGP propagation and best-path selection are used. At the headend of a specific policy, this will result in one or more candidate paths being installed into the "BGP table". These paths are then passed to the SRPM. The SRPM may compare them to candidate paths learned via other mechanisms and will choose one or more paths to be installed in the data plane. BGP itself does not install SR Policy candidate paths into the data plane.

This document defines a new BGP address family (SAFI). In UPDATE messages of that address family, the NLRI identifies an SR Policy Candidate Path while the attributes encode the segment lists and other details of that SR Policy Candidate Path.

While for simplicity we may write that BGP advertises an SR Policy, it has to be understood that BGP advertises a candidate path of an SR policy and that this SR Policy might have several other candidate paths provided via BGP (via an NLRI with a different distinguisher as defined in this document), PCEP, NETCONF, or local policy configuration.

Typically, a controller defines the set of policies and advertise them to policy head-end routers (typically ingress routers). The policy advertisement uses BGP extensions defined in this document. The policy advertisement is, in most but not all of the cases, tailored for a specific policy head-end. In this case, the advertisement may be sent on a BGP session to that head-end and not propagated any further.

Alternatively, a router (i.e., a BGP egress router) advertises SR Policies representing paths to itself. In this case, it is possible to send the policy to each head-end over a BGP session to that head-end, without requiring any further propagation of the policy.

An SR Policy intended only for the receiver will, in most cases, not traverse any Route Reflector (RR, [RFC4456]).

In some situations, it is undesirable for a controller or BGP egress router to have a BGP session to each policy head-end. In these

situations, BGP Route Reflectors may be used to propagate the advertisements, or it may be necessary for the advertisement to propagate through a sequence of one or more AS. To make this possible, an attribute needs to be attached to the advertisement that enables a BGP speaker to determine whether it is intended to be a head-end for the advertised policy. This is done by attaching one or more Route Target Extended Communities to the advertisement ([RFC4360]).

The BGP extensions for the advertisement of SR Policies include following components:

- o A new Subsequent Address Family Identifier (SAFI) whose NLRI identifies an SR Policy candidate path.
- o A new Tunnel Type identifier for SR Policy, and a set of sub-TLVs to be inserted into the Tunnel Encapsulation Attribute (as defined in [RFC9012]) specifying segment lists of the SR Policy candidate path, as well as other information about the SR Policy.
- o One or more IPv4 address format route target extended community ([RFC4360]) attached to the SR Policy advertisement and that indicates the intended head-end of such SR Policy advertisement.

The Color Extended Community (as defined in [RFC9012]) is used to steer traffic into an SR Policy, as described in section 8.8 of [I-D.ietf-spring-segment-routing-policy]. This document (Section 3) updates [RFC9012] with modifications to the format of the Color Extended Community by using the two leftmost bits of the RESERVED field.

1.1. Requirements Language

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

2. SR Policy Encoding

2.1. SR Policy SAFI and NLRI

A new SAFI is defined: the SR Policy SAFI with codepoint 73. The AFI used MUST be IPv4(1) or IPv6(2).

The SR Policy SAFI uses a new NLRI defined as follows:

NLRI Length	1 octet
Distinguisher	4 octets
Policy Color	4 octets
Endpoint	4 or 16 octets

where:

- o NLRI Length: 1 octet of length expressed in bits as defined in [RFC4760]. When AFI = 1 value MUST be 96 and when AFI = 2 value MUST be 192.
- o Distinguisher: 4-octet value uniquely identifying the policy in the context of <color, endpoint> tuple. The distinguisher has no semantic value and is solely used by the SR Policy originator to make unique (from an NLRI perspective) both for multiple candidate paths of the same SR Policy as well as candidate paths of different SR Policies (i.e. with different segment list) with the same Color and Endpoint but meant for different head-ends.
- o Policy Color: 4-octet value identifying (with the endpoint) the policy. The color is used to match the color of the destination prefixes to steer traffic into the SR Policy as specified in [I-D.ietf-spring-segment-routing-policy].
- o Endpoint: identifies the endpoint of a policy. The Endpoint may represent a single node or a set of nodes (e.g., an anycast address). The Endpoint is an IPv4 (4-octet) address or an IPv6 (16-octet) address according to the AFI of the NLRI.

The color and endpoint are used to automate the steering of BGP Payload prefixes on SR Policy as described in [I-D.ietf-spring-segment-routing-policy].

The NLRI containing the SR Policy candidate path is carried in a BGP UPDATE message [RFC4271] using BGP multi-protocol extensions [RFC4760] with an AFI of 1 or 2 (IPv4 or IPv6) and with a SAFI of 73.

An update message that carries the MP_REACH_NLRI or MP_UNREACH_NLRI attribute with the SR Policy SAFI MUST also carry the BGP mandatory attributes. In addition, the BGP update message MAY also contain any of the BGP optional attributes.

The next-hop network address field in SR Policy SAFI (73) updates may be either a 4 octet IPv4 address or a 16 octet IPv6 address, independent of the SR Policy AFI. The length field of the next-hop address specifies the next-hop address family. If the next-hop length is 4, then the next-hop is an IPv4 address; if the next-hop length is 16, then it is a global IPv6 address; if the next-hop length is 32, then it has a global IPv6 address followed by a link-local IPv6 address. The setting of the next-hop field and its attendant processing is governed by standard BGP procedures as described in section 3 in [RFC4760].

It is important to note that any BGP speaker receiving a BGP message with an SR Policy NLRI, will process it only if the NLRI is among the best-paths as per the BGP best-path selection algorithm. In other words, this document leverages the existing BGP propagation and best-path selection rules. Details of the procedures are described in Section 4.

It has to be noted that if several candidate paths of the same SR Policy (endpoint, color) are signaled via BGP to a head-end, it is RECOMMENDED that each NLRI uses a different distinguisher. If BGP has installed into the BGP table two advertisements whose respective NLRIs have the same color and endpoint, but different distinguishers, both advertisements are passed to the SRPM as different candidate paths along with their respective originator information (i.e. ASN and BGP Router-ID) as described in section 2.4 of [I-D.ietf-spring-segment-routing-policy]. The ASN would be the ASN of origin and the BGP Router-ID is determined in the following order:

- o From the Route Origin Community [RFC4360] if present and carrying an IP Address
- o As the BGP Originator ID [RFC4456] if present
- o As the BGP Router-ID of the peer from which the update was received as a last resort.

2.2. SR Policy and Tunnel Encapsulation Attribute

The content of the SR Policy Candidate Path is encoded in the Tunnel Encapsulation Attribute defined in [RFC9012] using a new Tunnel-Type called SR Policy Type with codepoint 15. The use of SR Policy Tunnel-type is applicable only for the AFI/SAFI pairs of (1/73, 2/73).

The SR Policy Encoding structure is as follows:

SR Policy SAFI NLRI: <Distinguisher, Policy-Color, Endpoint>

Attributes:

 Tunnel Encaps Attribute (23)

 Tunnel Type: SR Policy

 Binding SID

 SRv6 Binding SID

 Preference

 Priority

 Policy Name

 Policy Candidate Path Name

 Explicit NULL Label Policy (ENLP)

 Segment List

 Weight

 Segment

 Segment

 ...

 ...

where:

- o SR Policy SAFI NLRI is defined in Section 2.1.
- o Tunnel Encapsulation Attribute is defined in [RFC9012].
- o Tunnel-Type is set to 15.
- o Preference, Binding SID, SRv6 Binding SID, Priority, Policy Name, Policy Candidate Path Name, ENLP, Segment-List, Weight, and Segment sub-TLVs are defined in this document.
- o Additional sub-TLVs may be defined in the future.

A Tunnel Encapsulation Attribute MUST NOT contain more than one TLV of type "SR Policy".

2.3. Remote Endpoint and Color

The Tunnel Egress Endpoint and Color sub-TLVs, as defined in [RFC9012], may also be present in the SR Policy encodings.

The Tunnel Egress Endpoint and Color Sub-TLVs of the Tunnel Encapsulation Attribute are not used for SR Policy encodings and therefore their value is irrelevant in the context of the SR Policy SAFI NLRI. If present, the Tunnel Egress Endpoint sub-TLV and the Color sub-TLV MUST be ignored by the BGP speaker and not removed from the Tunnel Encapsulation Attribute during propagation.

2.4. SR Policy Sub-TLVs

This section specifies the sub-TLVs defined for encoding the information about the SR Policy Candidate Path.

Preference, Binding SID, SRv6 Binding SID, Segment-List, Priority, Policy Name, Policy Candidate Path Name, and Explicit NULL Label Policy are the new sub-TLVs of the BGP Tunnel Encapsulation Attribute [RFC9012] being defined in this section.

Weight and Segment are sub-TLVs of the new Segment-List sub-TLV mentioned above.

None of the sub-TLVs defined in the following sub-sections have any effect on the BGP best-path selection or propagation procedures. These sub-TLVs are not used by BGP and are instead passed on to SRPM as SR Policy Candidate Path information for further processing described in [I-D.ietf-spring-segment-routing-policy].

The use of SR Policy Sub-TLVs is applicable only for the AFI/SAFI pairs of (1/73, 2/73). Future documents may extend their applicability to other AFI/SAFI.

2.4.1. Preference Sub-TLV

The Preference sub-TLV is used to carry the preference of the SR Policy candidate path. The contents of this sub-TLV are used by the SRPM as described in section 2.7 in [I-D.ietf-spring-segment-routing-policy].

The Preference sub-TLV is optional and it MUST NOT appear more than once in the SR Policy encoding.

The Preference sub-TLV has following format:

0										1										2										3															
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1														
Type										Length										Flags										RESERVED															
Preference (4 octets)																																													

where:

- o Type: 12
- o Length: 6.

- o **Flags:** 1 octet of flags. None are defined at this stage. Flags SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o **RESERVED:** 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o **Preference:** a 4-octet value.

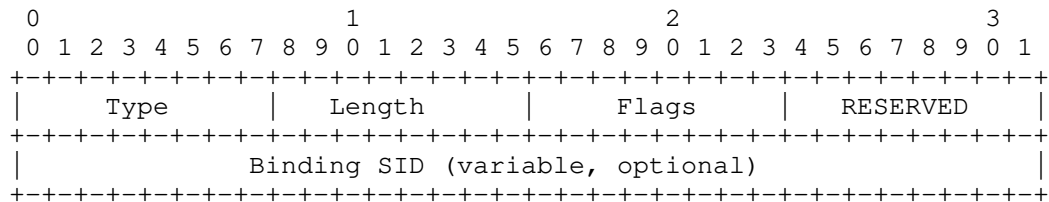
2.4.2. Binding SID Sub-TLV

The Binding SID sub-TLV is used to signal the binding SID related information of the SR Policy candidate path. The contents of this sub-TLV are used by the SRPM as described in section 6 in [I-D.ietf-spring-segment-routing-policy].

The Binding SID sub-TLV is optional and it MUST NOT appear more than once in the SR Policy encoding.

When the Binding SID sub-TLV is used to signal an SRv6 SID, the choice of its SRv6 Endpoint Behavior [RFC8986] to be instantiated is left to the headend node. It is RECOMMENDED that the SRv6 Binding SID sub-TLV defined in Section 2.4.3, that enables the specification of the SRv6 Endpoint Behavior, be used for signaling of an SRv6 Binding SID for an SR Policy candidate path.

The Binding SID sub-TLV has the following format:



where:

- o **Type:** 13
- o **Length:** specifies the length of the value field not including Type and Length fields. Can be 2 or 6 or 18.
- o **Flags:** 1 octet of flags. Following flags are defined in the new registry "SR Policy Binding SID Flags" as described in Section 6.6:

```

  0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|S|I|               |
+---+---+---+---+---+---+

```

where:

- * S-Flag: This flag encodes the "Specified-BSID-only" behavior. It is used by SRPM as described in section 6.2.3 in [I-D.ietf-spring-segment-routing-policy].
- * I-Flag: This flag encodes the "Drop Upon Invalid" behavior. It is used by SRPM as described in section 8.2 in [I-D.ietf-spring-segment-routing-policy].
- * Unused bits in the Flag octet SHOULD be set to zero upon transmission and MUST be ignored upon receipt.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Binding SID: if the length is 2, then no Binding SID is present. If the length is 6 then the Binding SID is encoded in 4 octets using the format below. TC, S, TTL (Total of 12 bits) are RESERVED and SHOULD be set to zero and MUST be ignored.

```

      0               1               2               3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               Label               | TC |S|               TTL       |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

If the length is 18 then the Binding SID contains a 16-octet SRv6 SID.

2.4.3. SRv6 Binding SID Sub-TLV

The SRv6 Binding SID sub-TLV is used to signal the SRv6 Binding SID related information of the SR Policy candidate path. It enables the specification of the SRv6 Endpoint Behavior [RFC8986] to be instantiated on the headend node. The contents of this sub-TLV are used by the SRPM as described in section 6 in [I-D.ietf-spring-segment-routing-policy].

The SRv6 Binding SID sub-TLV is optional. More than one SRv6 Binding SIDs MAY be signaled in the same SR Policy encoding to indicate one or more SRv6 SIDs, each with potentially different SRv6 Endpoint Behaviors to be instantiated.

The SRv6 Binding SID sub-TLV has the following format:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Type      |      Length      |      Flags      |  RESERVED  |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     SRv6 Binding SID (16 octets)                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
//      SRv6 Endpoint Behavior and SID Structure (optional)      //
+-----+-----+-----+-----+-----+-----+-----+-----+

```

where:

- o Type: TBD
- o Length is variable
- o Flags: 1 octet of flags. Following flags are defined in the new registry "SR Policy Binding SID Flags" as described in Section 6.7:

```

      0 1 2 3 4 5 6 7
+-----+-----+-----+-----+
|S|I|B|                                     |
+-----+-----+-----+-----+

```

where:

- * S-Flag: This flag encodes the "Specified-BSID-only" behavior. It is used by SRPM as described in section 6.2.3 in [I-D.ietf-spring-segment-routing-policy].
- * I-Flag: This flag encodes the "Drop Upon Invalid" behavior. It is used by SRPM as described in section 8.2 in [I-D.ietf-spring-segment-routing-policy].
- * B-Flag: This flag, when set, indicates the presence of the SRv6 Endpoint Behavior and SID Structure encoding specified in Section 2.4.4.2.13.
- * Unused bits in the Flag octet SHOULD be set to zero upon transmission and MUST be ignored upon receipt.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o SRv6 Binding SID: Contains a 16-octet SRv6 SID.

- o SRv6 Endpoint Behavior and SID Structure: Optional, as defined in Section 2.4.4.2.13.

2.4.4. Segment List Sub-TLV

The Segment List sub-TLV encodes a single explicit path towards the endpoint as described in section 5.1 in [I-D.ietf-spring-segment-routing-policy]. The Segment List sub-TLV includes the elements of the paths (i.e., segments) as well as an optional Weight sub-TLV.

The Segment List sub-TLV may exceed 255 bytes length due to large number of segments. Therefore a 2-octet length is required. According to [RFC9012], the first bit of the sub-TLV codepoint defines the size of the length field. Therefore, for the Segment List sub-TLV a code point of 128 or higher is used.

The Segment List sub-TLV is optional and MAY appear multiple times in the SR Policy encoding. The ordering of Segment List sub-TLVs, each sub-TLV encoding a Segment List, does not matter.

The Segment List sub-TLV contains zero or more Segment sub-TLVs and MAY contain a Weight sub-TLV.

The Segment List sub-TLV has the following format:

```

      0                               1                               2                               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Type      |                               Length                               | RESERVED |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//                               sub-TLVs                               //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

where:

- o Type: 128.
- o Length: the total length (not including the Type and Length fields) of the sub-TLVs encoded within the Segment List sub-TLV.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o sub-TLVs currently defined:
 - * An optional single Weight sub-TLV.

- * Zero or more Segment sub-TLVs.

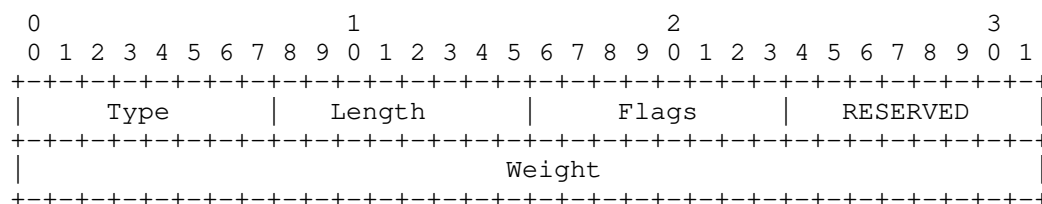
Validation of an explicit path encoded by the Segment List sub-TLV is beyond the scope of BGP and performed by the SRPM as described in section 5 in [I-D.ietf-spring-segment-routing-policy].

2.4.4.1. Weight Sub-TLV

The Weight sub-TLV specifies the weight associated with a given segment list. The contents of this sub-TLV are used only by the SRPM as described in section 2.11 in [I-D.ietf-spring-segment-routing-policy].

The Weight sub-TLV is optional and it MUST NOT appear more than once inside the Segment List sub-TLV.

The Weight sub-TLV has the following format:



where:

- o Type: 9.
- o Length: 6
- o Flags: 1 octet of flags. None are defined at this stage. Flags SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.

2.4.4.2. Segment Sub-TLVs

A Segment sub-TLV describes a single segment in a segment list (i.e., a single element of the explicit path). One or more Segment sub-TLVs constitute an explicit path of the SR Policy candidate path. The contents of these sub-TLVs are used only by the SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy].

The Segment sub-TLVs are optional and MAY appear multiple times in the Segment List sub-TLV.

[I-D.ietf-spring-segment-routing-policy] defines several Segment Types:

Type A: SR-MPLS Label
 Type B: SRv6 SID
 Type C: IPv4 Prefix with optional SR Algorithm
 Type D: IPv6 Global Prefix with optional SR Algorithm for SR-MPLS
 Type E: IPv4 Prefix with Local Interface ID
 Type F: IPv4 Addresses for link endpoints as Local, Remote pair
 Type G: IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SR-MPLS
 Type H: IPv6 Addresses for link endpoints as Local, Remote pair for SR-MPLS
 Type I: IPv6 Global Prefix with optional SR Algorithm for SRv6
 Type J: IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SRv6
 Type K: IPv6 Addresses for link endpoints as Local, Remote pair for SRv6

The following sub-sections specify the sub-TLV used for encoding each of these Segment Types.

2.4.4.2.1. Segment Type A

The Type A Segment Sub-TLV encodes a single SR-MPLS SID. The format is as follows:

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
Type										Length										Flags										RESERVED									
Label										TC										S										TTL									

where:

- o Type: 1.
- o Length is 6.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.

- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Label: 20 bits of label value.
- o TC: 3 bits of traffic class.
- o S: 1 bit of bottom-of-stack.
- o TTL: 1 octet of TTL.

The following applies to the Type-1 Segment sub-TLV:

- o The S bit SHOULD be zero upon transmission and MUST be ignored upon reception.
- o If the originator wants the receiver to choose the TC value, it sets the TC field to zero.
- o If the originator wants the receiver to choose the TTL value, it sets the TTL field to 255.
- o If the originator wants to recommend a value for these fields, it puts those values in the TC and/or TTL fields.
- o The receiver MAY override the originator's values for these fields. This would be determined by local policy at the receiver. One possible policy would be to override the fields only if the fields have the default values specified above.

2.4.4.2.2. Segment Type B

The Type B Segment Sub-TLV encodes a single SRv6 SID. The format is as follows:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Type      |    Length    |    Flags    |  RESERVED  |
+-----+-----+-----+-----+-----+-----+-----+-----+
//                SRv6 SID (16 octets)                //
+-----+-----+-----+-----+-----+-----+-----+-----+
//      SRv6 Endpoint Behavior and SID Structure (optional)      //
+-----+-----+-----+-----+-----+-----+-----+-----+

```

where:

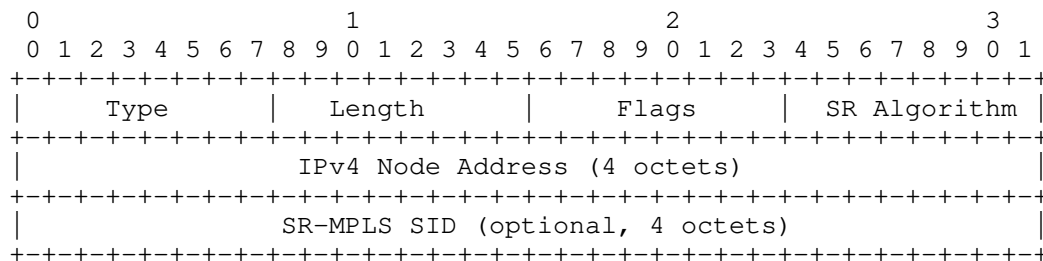
- o Type: 13.

- o Length is variable.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o SRv6 SID: 16 octets of IPv6 address.
- o SRv6 Endpoint Behavior and SID Structure: Optional, as defined in Section 2.4.4.2.13.

The TLV 2 defined for the advertisement of Segment Type B in the earlier versions of this document has been deprecated to avoid backward compatibility issues.

2.4.4.2.3. Segment Type C

The Type C Segment Sub-TLV encodes an IPv4 node address, SR Algorithm and an optional SR-MPLS SID. The format is as follows:



where:

- o Type: 3.
- o Length is 10 when the SR-MPLS SID is present else is 6.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o SR Algorithm: 1 octet specifying SR Algorithm as described in section 3.1.1 in [RFC8402] when A-Flag as defined in Section 2.4.4.2.12 is present. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy]. When A-Flag is not encoded, this field SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o IPv4 Node Address: a 4 octet IPv4 address representing a node.

- o SR-MPLS SID: optional, 4 octet field containing label, TC, S and TTL as defined in Section 2.4.4.2.1.

2.4.4.2.4. Segment Type D

The Type D Segment Sub-TLV encodes an IPv6 node address, SR Algorithm and an optional SR-MPLS SID. The format is as follows:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|          Type          |      Length      |      Flags      | SR Algorithm |
+-----+-----+-----+-----+-----+-----+-----+-----+
//          IPv6 Node Address (16 octets)          //
+-----+-----+-----+-----+-----+-----+-----+-----+
|          SR-MPLS SID (optional, 4 octets)          |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

where:

- o Type: 4
- o Length is 22 when the SR-MPLS SID is present else is 18.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o SR Algorithm: 1 octet specifying SR Algorithm as described in section 3.1.1 in [RFC8402] when A-Flag as defined in Section 2.4.4.2.12 is present. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy]. When A-Flag is not encoded, this field SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o IPv6 Node Address: a 16 octet IPv6 address representing a node.
- o SR-MPLS SID: optional, 4 octet field containing label, TC, S and TTL as defined in Section 2.4.4.2.1.

2.4.4.2.5. Segment Type E

The Type E Segment Sub-TLV encodes an IPv4 node address, a local interface Identifier (Local Interface ID), and an optional SR-MPLS SID. The format is as follows:

0	1	2	3
0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1
-----	-----	-----	-----
Type	Length	Flags	RESERVED
-----	-----	-----	-----
	Local Interface ID (4 octets)		
-----	-----	-----	-----
	IPv4 Node Address (4 octets)		
-----	-----	-----	-----
	SR-MPLS SID (optional, 4 octets)		
-----	-----	-----	-----

where:

- o Type: 5.
- o Length is 14 when the SR-MPLS SID is present else is 10.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Local Interface ID: 4 octets of interface index as defined in [RFC8664].
- o IPv4 Node Address: a 4 octet IPv4 address representing a node.
- o SR-MPLS SID: optional, 4 octet field containing label, TC, S and TTL as defined in Section 2.4.4.2.1.

2.4.4.2.6. Segment Type F

The Type F Segment Sub-TLV encodes an adjacency local address, an adjacency remote address, and an optional SR-MPLS SID. The format is as follows:

0	1	2	3
0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1
-----	-----	-----	-----
Type	Length	Flags	RESERVED
-----	-----	-----	-----
	Local IPv4 Address (4 octets)		
-----	-----	-----	-----
	Remote IPv4 Address (4 octets)		
-----	-----	-----	-----
	SR-MPLS SID (optional, 4 octets)		
-----	-----	-----	-----

where:

- o Type: 6.
- o Length is 14 when the SR-MPLS SID is present else is 10.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Local IPv4 Address: a 4 octet IPv4 address.
- o Remote IPv4 Address: a 4 octet IPv4 address.
- o SR-MPLS SID: optional, 4 octet field containing label, TC, S and TTL as defined in Section 2.4.4.2.1.

2.4.4.2.7. Segment Type G

The Type G Segment Sub-TLV encodes an IPv6 link-local adjacency with IPv6 local node address, a local interface identifier (Local Interface ID), IPv6 remote node address, a remote interface identifier (Remote Interface ID), and an optional SR-MPLS SID. The format is as follows:

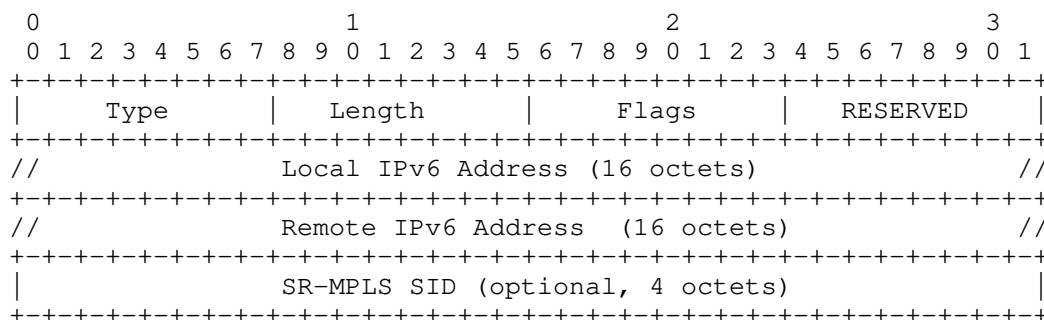
0								1								2								3							
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Type								Length								Flags								RESERVED							
								Local Interface ID (4 octets)																							
//								IPv6 Local Node Address (16 octets)																//							
								Remote Interface ID (4 octets)																							
//								IPv6 Remote Node Address (16 octets)																//							
								SR-MPLS SID (optional, 4 octets)																							

where:

- o Type: 7
- o Length is 46 when the SR-MPLS SID is present else is 42.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Local Interface ID: 4 octets of interface index as defined in [RFC8664].
- o IPv6 Local Node Address: a 16 octet IPv6 address.
- o Remote Interface ID: 4 octets of interface index as defined in [RFC8664]. The value MAY be set to zero when the local node address and interface identifiers are sufficient to describe the link.
- o IPv6 Remote Node Address: a 16 octet IPv6 address. The value MAY be set to zero when the local node address and interface identifiers are sufficient to describe the link.
- o SR-MPLS SID: optional, 4 octet field containing label, TC, S and TTL as defined in Section 2.4.4.2.1.

2.4.4.2.8. Segment Type H

The Type H Segment Sub-TLV encodes an adjacency local address, an adjacency remote address, and an optional SR-MPLS SID. The format is as follows:



where:

- o Type: 8
- o Length is 38 when the SR-MPLS SID is present else is 34.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Local IPv6 Address: a 16 octet IPv6 address.
- o Remote IPv6 Address: a 16 octet IPv6 address.
- o SR-MPLS SID: optional, 4 octet field containing label, TC, S and TTL as defined in Section 2.4.4.2.1.

2.4.4.2.9. Segment Type I

The Type I Segment Sub-TLV encodes an IPv6 node address, SR Algorithm, and an optional SRv6 SID. The format is as follows:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Type   |   Length   |   Flags   | SR Algorithm |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//          IPv6 Node Address (16 octets)          //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//          SRv6 SID (optional, 16 octets)          //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//          SRv6 Endpoint Behavior and SID Structure (optional) //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

where:

- o Type: 14
- o Length is variable.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o SR Algorithm: 1 octet specifying SR Algorithm as described in section 3.1.1 in [RFC8402] when A-Flag as defined in Section 2.4.4.2.12 is present. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy]. When A-Flag is not encoded, this field SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o IPv6 Node Address: a 16 octet IPv6 address.
- o SRv6 SID: optional, a 16 octet IPv6 address.
- o SRv6 Endpoint Behavior and SID Structure: Optional, as defined in Section 2.4.4.2.13.

The TLV 10 defined for the advertisement of Segment Type I in the earlier versions of this document has been deprecated to avoid backward compatibility issues.

2.4.4.2.10. Segment Type J

The Type J Segment Sub-TLV encodes an IPv6 link-local adjacency with local node address, a local interface identifier (Local Interface ID), remote IPv6 node address, a remote interface identifier (Remote Interface ID), and an optional SRv6 SID. The format is as follows:


```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Type      |      Length      |      Flags      | SR Algorithm |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     Local Interface ID (4 octets) |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//                                     IPv6 Local Node Address (16 octets) //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     Remote Interface ID (4 octets) |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//                                     IPv6 Remote Node Address (16 octets) //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//                                     SRv6 SID (optional, 16 octets) //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//          SRv6 Endpoint Behavior and SID Structure (optional) //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

where:

- o Type: 15
- o Length is variable.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o SR Algorithm: 1 octet specifying SR Algorithm as described in section 3.1.1 in [RFC8402] when A-Flag as defined in Section 2.4.4.2.12 is present. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy]. When A-Flag is not encoded, this field SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Local Interface ID: 4 octets of interface index as defined in [RFC8664].
- o IPv6 Local Node Address: a 16 octet IPv6 address.
- o Remote Interface ID: 4 octets of interface index as defined in [RFC8664]. The value MAY be set to zero when the local node address and interface identifiers are sufficient to describe the link.
- o IPv6 Remote Node Address: a 16 octet IPv6 address. The value MAY be set to zero when the local node address and interface identifiers are sufficient to describe the link.

- o SRv6 SID: optional, a 16 octet IPv6 address.
- o SRv6 Endpoint Behavior and SID Structure: Optional, as defined in Section 2.4.4.2.13.

The TLV 11 defined for the advertisement of Segment Type J in the earlier versions of this document has been deprecated to avoid backward compatibility issues.

2.4.4.2.11. Segment Type K

The Type K Segment Sub-TLV encodes an adjacency local address, an adjacency remote address, and an optional SRv6 SID. The format is as follows:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|   Type   |   Length   |   Flags   | SR Algorithm |
+-----+-----+-----+-----+-----+-----+-----+-----+
//          Local IPv6 Address (16 octets)          //
+-----+-----+-----+-----+-----+-----+-----+-----+
//          Remote IPv6 Address (16 octets)          //
+-----+-----+-----+-----+-----+-----+-----+-----+
//          SRv6 SID (optional, 16 octets)           //
+-----+-----+-----+-----+-----+-----+-----+-----+
//          SRv6 Endpoint Behavior and SID Structure (optional) //
+-----+-----+-----+-----+-----+-----+-----+-----+

```

where:

- o Type: 16
- o Length is variable.
- o Flags: 1 octet of flags as defined in Section 2.4.4.2.12.
- o SR Algorithm: 1 octet specifying SR Algorithm as described in section 3.1.1 in [RFC8402] when A-Flag as defined in Section 2.4.4.2.12 is present. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy]. When A-Flag is not encoded, this field SHOULD be set to zero on transmission and MUST be ignored on receipt.
- o Local IPv6 Address: a 16 octet IPv6 address.
- o Remote IPv6 Address: a 16 octet IPv6 address.

- o SRv6 SID: optional, a 16 octet IPv6 address.
- o SRv6 Endpoint Behavior and SID Structure: Optional, as defined in Section 2.4.4.2.13.

The TLV 12 defined for the advertisement of Segment Type K in the earlier versions of this document has been deprecated to avoid backward compatibility issues.

2.4.4.2.12. Segment Flags

The Segment Types sub-TLVs described above MAY contain the following flags in the "Flags" field defined in Section 6.8:

```

 0 1 2 3 4 5 6 7
+---+---+---+---+
|V|A|S|B|   |
+---+---+---+---+
```

where:

V-Flag: This flag, when set, is used by SRPM for "SID verification" as described in Section 5.1 in [I-D.ietf-spring-segment-routing-policy].

A-Flag: This flag, when set, indicates the presence of SR Algorithm id in the "SR Algorithm" field applicable to various Segment Types. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy].

S-Flag: This flag, when set, indicates the presence of the SR-MPLS or SRv6 SID depending on the segment type.

B-Flag: This flag, when set, indicates the presence of the SRv6 Endpoint Behavior and SID Structure encoding specified in Section 2.4.4.2.13.

Unused bits in the Flag octet SHOULD be set to zero upon transmission and MUST be ignored upon receipt.

The following applies to the Segment Flags:

- o V-Flag applies to all Segment Types.
- o A-Flag applies to Segment Types C, D, I, J, and K. If A-Flag appears with Segment Types A, B, E, F, G, and H, it MUST be ignored.

- o S-Flag applies to Segment Types C, D, E, F, G, H, I, J, and K. If S-Flag appears with Segment Types A or B, it MUST be ignored.
- o B-Flag applies to Segment Types B, I, J, and K. If B-Flag appears with Segment Types A, C, D, E, F, G, and H, it MUST be ignored.

2.4.4.2.13. SRv6 SID Endpoint Behavior and Structure

The Segment Types sub-TLVs described above MAY contain the SRv6 Endpoint Behavior and SID Structure [RFC8986] encoding as described below:

```

+-----+
|           Endpoint Behavior           |           Reserved           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|   LB Length   |   LN Length   | Fun. Length   | Arg. Length   |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

where:

Endpoint Behavior: 2 octets. It carries the SRv6 Endpoint Behavior code point for this SRv6 SID as defined in section 9.2 of [RFC8986]. When set with the value 0, the choice of SRv6 Endpoint Behavior is left to the headend.

Reserved: 2 octets of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.

Locator Block Length: 1 octet. SRv6 SID Locator Block length in bits.

Locator Node Length: 1 octet. SRv6 SID Locator Node length in bits.

Function Length: 1 octet. SRv6 SID Function length in bits.

Argument Length: 1 octet. SRv6 SID Arguments length in bits.

The total of the locator block, locator node, function, and argument lengths MUST be less than or equal to 128.

2.4.5. Explicit NULL Label Policy Sub-TLV

To steer an unlabeled IP packet into an SR policy, it is necessary to create a label stack for that packet, and push one or more labels onto that stack.

The Explicit NULL Label Policy (ENLP) sub-TLV is used to indicate whether an Explicit NULL Label [RFC3032] must be pushed on an unlabeled IP packet before any other labels.

If an ENLP Sub-TLV is not present, the decision of whether to push an Explicit NULL label on a given packet is a matter of local configuration.

The ENLP sub-TLV is optional and it MUST NOT appear more than once in the SR Policy encoding.

The contents of this sub-TLV are used by the SRPM as described in section 4.1 in [I-D.ietf-spring-segment-routing-policy].

0								1								2								3							
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Type								Length								Flags								RESERVED							
ENLP																															

Where:

Type: 14.

Length: 3.

Flags: 1 octet of flags. None are defined at this stage. Flags SHOULD be set to zero on transmission and MUST be ignored on receipt.

RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.

ENLP (Explicit NULL Label Policy): Indicates whether Explicit NULL labels are to be pushed on unlabeled IP packets that are being steered into a given SR policy. This field has one of the following values:

0: Reserved.

1: Push an IPv4 Explicit NULL label on an unlabeled IPv4 packet, but do not push an IPv6 Explicit NULL label on an unlabeled IPv6 packet.

2: Push an IPv6 Explicit NULL label on an unlabeled IPv6 packet, but do not push an IPv4 Explicit NULL label on an unlabeled IPv4 packet.

3: Push an IPv4 Explicit NULL label on an unlabeled IPv4 packet, and push an IPv6 Explicit NULL label on an unlabeled IPv6 packet.

4: Do not push an Explicit NULL label.

5 - 255: Reserved.

The ENLP reserved values may be used for future extensions and implementations SHOULD ignore the ENLP Sub-TLV with these values. The behavior signaled in this Sub-TLV MAY be overridden by local configuration. The section 4.1 of [I-D.ietf-spring-segment-routing-policy] describes the behavior on the headend for the handling of the explicit null label.

2.4.6. Policy Priority Sub-TLV

An operator MAY set the Policy Priority sub-TLV to indicate the order in which the SR policies are re-computed upon topological change. The contents of this sub-TLV are used by the SRPM as described in section 2.11 in [I-D.ietf-spring-segment-routing-policy].

The Priority sub-TLV is optional and it MUST NOT appear more than once in the SR Policy encoding.

The Priority sub-TLV has following format:

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
Type										Length										Priority										RESERVED									

Where:

Type: 15

Length: 2.

Priority: a 1-octet value.

RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.

2.4.7. Policy Candidate Path Name Sub-TLV

An operator MAY set the Policy Candidate Path Name sub-TLV to attach a symbolic name to the SR Policy candidate path.

Usage of Policy Candidate Path Name sub-TLV is described in section 2.6 in [I-D.ietf-spring-segment-routing-policy].

The Policy Candidate Path Name sub-TLV may exceed 255 bytes length due to a long name. Therefore a 2-octet length is required. According to [RFC9012], the first bit of the sub-TLV codepoint defines the size of the length field. Therefore, for the Policy Candidate Path Name sub-TLV, a code point of 128 or higher is used.

It is RECOMMENDED that the size of the symbolic name for the candidate path be limited to 255 bytes. Implementations MAY choose to truncate long names to 255 bytes when signaling via BGP.

The Policy Candidate Path Name sub-TLV is optional and it MUST NOT appear more than once in the SR Policy encoding.

The Policy Candidate Path Name sub-TLV has following format:

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Type      |      Length      |      RESERVED      |
+-----+-----+-----+-----+-----+-----+-----+
//              Policy Candidate Path Name              //
```

Where:

Type: 129.

Length: Variable.

RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.

Policy Candidate Path Name: Symbolic name for the SR Policy candidate path without a NULL terminator as specified in section 2.6 of [I-D.ietf-spring-segment-routing-policy].

2.4.8. Policy Name Sub-TLV

An operator MAY set the Policy Name sub-TLV to associate a symbolic name with the SR Policy for which the candidate path is being advertised via the SR Policy NLRI.

Usage of Policy Name sub-TLV is described in section 2.1 of [I-D.ietf-spring-segment-routing-policy].

The Policy Name sub-TLV may exceed 255 bytes length due to a long policy name. Therefore a 2-octet length is required. According to [RFC9012], the first bit of the sub-TLV codepoint defines the size of the length field. Therefore, for the Policy Name sub-TLV, a code point of 128 or higher is used.

It is RECOMMENDED that the size of the symbolic name for the SR Policy be limited to 255 bytes. Implementations MAY choose to truncate long names to 255 bytes when signaling via BGP.

The Policy Name sub-TLV is optional and it MUST NOT appear more than once in the SR Policy encoding.

The Policy Name sub-TLV has following format:

```

0                               1                               2                               3
0 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Type      |      Length      |      RESERVED      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//                               Policy Name                               //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Where:

Type: TBD

Length: Variable.

RESERVED: 1 octet of reserved bits. SHOULD be set to zero on transmission and MUST be ignored on receipt.

Policy Name: Symbolic name for the policy. It SHOULD be a string of printable ASCII characters, without a NULL terminator.

3. Color Extended Community

The Color Extended Community [RFC9012] is used to steer traffic corresponding to BGP routes (e.g., L3VPN) into an SR Policy with matching color value.

Two bits from the Flags field of the Color Extended Community are used as follows to support the requirements of Color-Only steering as specified in Section 8.8 of [I-D.ietf-spring-segment-routing-policy]:

```

                                1
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+---+---+---+---+---+---+---+---+---+
|C O|           RESERVED           |
+---+---+---+---+---+---+---+---+

```

The CO bits together form the Color-Only Type field which indicates the various matching criteria between BGP NH and SR Policy endpoint in addition to the matching of the color value. Following types are defined:

- o Type 0: Specific Endpoint Match: Request match for the endpoint that is the BGP NH
- o Type 1: Specific or Null Endpoint Match: Request match for either the endpoint that is the BGP NH or a null endpoint (e.g., like a default gateway)
- o Type 2: Specific, Null or Any Endpoint Match: Request match for either the endpoint that is the BGP NH or with a null or any endpoint
- o Type 3: reserved for future use and SHOULD NOT be used. Upon reception, an implementation MUST treat it like Type 0.

The details of the SR Policy steering mechanisms based on these Color-Only types are specified in section 8.8 of [I-D.ietf-spring-segment-routing-policy].

One or more Color Extended Communities MAY be associated with a BGP route update. Sections 8.4.1, 8.5.1, and 8.8.2 of [I-D.ietf-spring-segment-routing-policy] specify the steering behaviors over SR Policies when multiple Color Extended Communities are associated with a BGP route.

4. SR Policy Operations

As described in this document, BGP is not the actual consumer of an SR Policy NLRI. BGP is in charge of the origination and propagation of the SR Policy NLRI but its installation and use are outside the scope of BGP. The details of SR Policy installation and use are specified in [I-D.ietf-spring-segment-routing-policy].

4.1. Advertisement of SR Policies

Typically, but not limited to, an SR Policy is computed by a controller or a path computation engine (PCE) and originated by a BGP speaker on its behalf.

Multiple SR Policy NLRIs may be present with the same <color, endpoint> tuple but with different content when these SR policies are intended for different head-ends.

The distinguisher of each SR Policy NLRI prevents undesired BGP route selection among these SR Policy NLRIs and allows their propagation across route reflectors [RFC4456].

Moreover, one or more route target SHOULD be attached to the advertisement, where each route target identifies one or more intended head-ends for the advertised SR Policy update.

If no route target is attached to the SR Policy NLRI, then it is assumed that the originator sends the SR Policy update directly (e.g., through a BGP session) to the intended receiver. In such case, the NO_ADVERTISE community MUST be attached to the SR Policy update.

4.2. Reception of an SR Policy NLRI

On reception of an SR Policy NLRI, a BGP speaker first determines if it is acceptable and then if it is usable.

4.2.1. Acceptance of an SR Policy NLRI

When a BGP speaker receives an SR Policy NLRI from a neighbor it MUST first, determine if it's acceptable. The following rules apply in addition to the validation described in Section 5:

- o The SR Policy NLRI MUST include a distinguisher, color and endpoint field which implies that the length of the NLRI MUST be either 12 or 24 octets (depending on the address family of the endpoint).

- o The SR Policy update MUST have either the NO_ADVERTISE community or at least one route target extended community in IPv4-address format or both. If a router supporting this specification receives an SR Policy update with no route target extended communities and no NO_ADVERTISE community, the update MUST be considered as malformed.
- o The Tunnel Encapsulation Attribute MUST be attached to the BGP Update and MUST have a Tunnel Type TLV set to SR Policy (codepoint is 15).

A router that receives an SR Policy update that is not valid according to these criteria MUST treat the update as malformed and the SR Policy candidate path MUST NOT be passed to the SRPM.

4.2.2. Usable SR Policy NLRI

An SR Policy update that has been determined to be acceptable is further evaluated for its usability by the receiving node.

An SR Policy NLRI update without any route target extended community but having the NO_ADVERTISE community is considered usable.

If one or more route targets are present, then at least one route target MUST match the BGP Identifier of the receiver for the update to be considered usable. The BGP Identifier is defined in [RFC4271] as a 4 octet IPv4 address. Therefore, the route target extended community MUST be of the same format.

If one or more route targets are present and none matches the local BGP Identifier, then, while the SR Policy NLRI is acceptable, it is not usable on the receiver node.

When the SR Policy tunnel type includes any sub-TLV that is unrecognized or unsupported, the update SHOULD NOT be considered usable. An implementation MAY provide an option for ignoring unsupported sub-TLVs.

4.2.3. Passing a usable SR Policy NLRI to the SRPM

Once BGP on the receiving node has determined that the SR Policy NLRI is usable, it passes the SR Policy candidate path to the SRPM. Note that, along with the candidate path details, BGP also passes the originator information for breaking ties in the candidate path selection process as described in section 2.4 in [I-D.ietf-spring-segment-routing-policy].

When an update for an SR Policy NLRI results in its becoming unusable, BGP MUST delete its corresponding SR Policy candidate path from the SRPM.

The SRPM applies the rules defined in section 2 in [I-D.ietf-spring-segment-routing-policy] to determine whether the SR Policy candidate path is valid and to select the best candidate path among the valid ones for a given SR Policy.

4.2.4. Propagation of an SR Policy

SR Policy NLRIs that have been determined acceptable and valid can be evaluated for propagation, even the ones that are not usable.

SR Policy NLRIs that have the NO_ADVERTISE community attached to them MUST NOT be propagated.

By default, a BGP node receiving an SR Policy NLRI MUST NOT propagate it to any EBGp neighbor. An implementation MAY provide an explicit configuration to override this and enable propagation of acceptable SR Policy NLRIs to specific EBGp neighbors.

A BGP node advertises a received SR Policy NLRI to its IBGP neighbors according to normal IBGP propagation rules.

By default, a BGP node receiving an SR Policy NLRI SHOULD NOT remove route target extended community before propagation. An implementation MAY provide support for configuration to filter and/or remove route target extended community before propagation.

5. Error Handling

This section describes the error handling actions, as described in [RFC7606], that are to be performed for the handling of BGP update messages for BGP SR Policy SAFI.

A BGP Speaker MUST perform the following syntactic validation of the SR Policy NLRI to determine if it is malformed. This includes the validation of the length of each NLRI and the total length of the MP_REACH_NLRI and MP_UNREACH_NLRI attributes.

When the error determined allows for the router to skip the malformed NLRI(s) and continue the processing of the rest of the update message, then it MUST handle such malformed NLRIs as 'Treat-as-withdraw'. In other cases, where the error in the NLRI encoding results in the inability to process the BGP update message (e.g. length related encoding errors), then the router SHOULD handle such malformed NLRIs as 'AFI/SAFI disable' when other AFI/SAFI besides SR

Policy are being advertised over the same session. Alternately, the router MUST perform 'session reset' when the session is only being used for SR Policy or when it 'AFI/SAFI disable' action is not possible.

The validation of the TLVs/sub-TLVs introduced in this document and defined in their respective sub-sections of Section 2.4 MUST be performed to determine if they are malformed or invalid. The validation of the Tunnel Encapsulation Attribute itself and the other TLVs/sub-TLVs specified in [RFC9012] MUST be done as described in that document. In case of any error detected, either at the attribute or its TLV/sub-TLV level, the "treat-as-withdraw" strategy MUST be applied. This is because an SR Policy update without a valid Tunnel Encapsulation Attribute (comprising of all valid TLVs/sub-TLVs) is not usable.

An SR Policy update that is determined to be not acceptable, and therefore malformed, based on rules described in Section 4.2.1 MUST be handled by the "treat-as-withdraw" strategy.

The validation of the individual fields of the TLVs/sub-TLVs defined in Section 2.4 are beyond the scope of BGP as they are handled by the SRPM as described in the individual TLV/sub-TLV sub-sections. A BGP implementation MUST NOT perform semantic verification of such fields nor consider the SR Policy update to be invalid or not acceptable/usable based on such validation.

An implementation SHOULD log an error for any errors found during the above validation for further analysis.

6. IANA Considerations

This document requests codepoint allocations in the following existing registries:

- o Subsequent Address Family Identifiers (SAFI) Parameters registry
- o BGP Tunnel Encapsulation Attribute Tunnel Types registry under the BGP Tunnel Encapsulation registry
- o BGP Tunnel Encapsulation Attribute sub-TLVs registry under the BGP Tunnel Encapsulation registry
- o Color Extended Community Flags registry under the BGP Tunnel Encapsulation registry

This document also requests the creation of the following new registries:

- o SR Policy Segment List Sub-TLVs under the BGP Tunnel Encapsulation registry
- o SR Policy Binding SID Flags under the BGP Tunnel Encapsulation registry
- o SR Policy Segment Flags under the BGP Tunnel Encapsulation registry
- o Color Extended Community Color-Only Types registry under the BGP Tunnel Encapsulation registry

6.1. Existing Registry: Subsequent Address Family Identifiers (SAFI) Parameters

This document defines a new SAFI in the registry "Subsequent Address Family Identifiers (SAFI) Parameters" that has been assigned a codepoint by IANA as follows:

Codepoint	Description	Reference
73	SR Policy SAFI	This document

6.2. Existing Registry: BGP Tunnel Encapsulation Attribute Tunnel Types

This document defines a new Tunnel-Type in the registry "BGP Tunnel Encapsulation Attribute Tunnel Types" that has been assigned a codepoint by IANA as follows:

Codepoint	Description	Reference
15	SR Policy	This document

6.3. Existing Registry: BGP Tunnel Encapsulation Attribute sub-TLVs

This document defines new sub-TLVs in the registry "BGP Tunnel Encapsulation Attribute sub-TLVs" that has been assigned codepoints by IANA as follows via the early allocation process:

Codepoint	Description	Reference
12	Preference sub-TLV	This document
13	Binding SID sub-TLV	This document
14	ENLP sub-TLV	This document
15	Priority sub-TLV	This document
20	SRv6 Binding SID sub-TLV	This document
128	Segment List sub-TLV	This document
129	Policy Candidate Path Name sub-TLV	This document
130	Policy Name sub-TLV	This document

6.4. Existing Registry: Color Extended Community Flags

This document requests allocations in the registry called "Color Extended Community Flags" under the "BGP Tunnel Encapsulation" registry.

The following bits have been assigned by IANA via the early allocation process to form the Color-Only Types field:

Bit Position	Description	Reference
0-1	Color-only Types Field	This document

6.5. New Registry: SR Policy Segment List Sub-TLVs

This document requests the creation of a new registry called "SR Policy Segment List Sub-TLVs" under the "BGP Tunnel Encapsulation" registry. The allocation policy of this registry is "Standards Action" according to [RFC8126].

Following initial Sub-TLV codepoints are assigned by this document:

Value	Description	Reference
0	Reserved	This document
1	Segment Type A sub-TLV	This document
2	Deprecated	This document
3	Segment Type C sub-TLV	This document
4	Segment Type D sub-TLV	This document
5	Segment Type E sub-TLV	This document
6	Segment Type F sub-TLV	This document
7	Segment Type G sub-TLV	This document
8	Segment Type H sub-TLV	This document
9	Weight sub-TLV	This document
10	Deprecated	This document
11	Deprecated	This document
12	Deprecated	This document
13	Segment Type B sub-TLV	This document
14	Segment Type I sub-TLV	This document
15	Segment Type J sub-TLV	This document
16	Segment Type K sub-TLV	This document
17-255	Unassigned	

6.6. New Registry: SR Policy Binding SID Flags

This document requests the creation of a new registry called "SR Policy Binding SID Flags" under the "BGP Tunnel Encapsulation" registry. The allocation policy of this registry is "Standards Action" according to [RFC8126].

The following flags are defined:

Bit	Description	Reference
0	Specified-BSID-Only Flag (S-Flag)	This document
1	Drop Upon Invalid Flag (I-Flag)	This document
2-7	Unassigned	

6.7. New Registry: SR Policy SRv6 Binding SID Flags

This document requests the creation of a new registry called "SR Policy SRv6 Binding SID Flags" under the "BGP Tunnel Encapsulation" registry. The allocation policy of this registry is "Standards Action" according to [RFC8126].

The following flags are defined:

Bit	Description	Reference
0	Specified-BSID-Only Flag (S-Flag)	This document
1	Drop Upon Invalid Flag (I-Flag)	This document
2	SRv6 Endpoint Behavior & SID Structure Flag (B-Flag)	This document
3-7	Unassigned	

6.8. New Registry: SR Policy Segment Flags

This document requests the creation of a new registry called "SR Policy Segment Flags" under the "BGP Tunnel Encapsulation" registry. The allocation policy of this registry is "Standards Action" according to [RFC8126].

The following Flags are defined:

Bit	Description	Reference
0	Segment Verification Flag (V-Flag)	This document
1	SR Algorithm Flag (A-Flag)	This document
2	SID Specified Flag (S-Flag)	This document
3	SRv6 Endpoint Behavior & SID Structure Flag (B-Flag)	This document
4-7	Unassigned	

6.9. New Registry: Color Extended Community Color-Only Types

This document requests the creation of a new registry called "Color Extended Community Color-Only Types" under the "BGP Tunnel Encapsulation" registry for assignment of codepoints (values 0 through 3) in the Color-Only Type field of the Color Extended Community Flags field. The allocation policy of this registry is "Standards Action" according to [RFC8126].

The following types are defined:

Type	Description	Reference
0	Specific Endpoint Match	This document
1	Specific or Null Endpoint Match	This document
2	Specific, Null or Any Endpoint Match	This document
3	Unallocated & reserved for future	This document

7. Security Considerations

The security mechanisms of the base BGP security model apply to the extensions described in this document as well. See the Security Considerations section of [RFC4271] for a discussion of BGP security. Also, refer to [RFC4272] and [RFC6952] for analysis of security issues for BGP.

The BGP SR Policy extensions specified in this document enable traffic engineering and service programming use-cases within the SR domain as described in [I-D.ietf-spring-segment-routing-policy]. SR operates within a trusted SR domain [RFC8402] and its security considerations also apply to BGP sessions when carrying SR Policy information. The SR Policies distributed by BGP are expected to be used entirely within this trusted SR domain i.e. within a single AS or between multiple AS/domains within a single provider network. Therefore, precaution is necessary to ensure that the SR Policy information advertised via BGP sessions is limited to nodes in a secure manner within this trusted SR domain. BGP peering sessions for address-families other than SR Policy SAFI may be set up to routers outside the SR domain. The isolation of BGP SR Policy SAFI peering sessions may be used to ensure that the SR Policy information is not advertised by accident or error to an EBGP peering session outside the SR domain.

Additionally, it may be considered that the export of SR Policy information, as described in this document, constitutes a risk to confidentiality of mission-critical or commercially sensitive information about the network (more specifically endpoint/node addresses, SR SIDs, and the SR Policies deployed). BGP peerings are not automatic and require configuration; thus, it is the responsibility of the network operator to ensure that only trusted nodes (that include both routers and controller applications) within the SR domain are configured to receive such information.

8. Acknowledgments

The authors of this document would like to thank Shyam Sethuram, John Scudder, Przemyslaw Krol, Alex Bogdanov, Nandan Saha, Bruno Decraene, Gurusiddesh Nidasesi, Kausik Majumdar, Zafar Ali, Swadesh Agarwal, Jakob Heitz, Viral Patel, Peng Shaofu, Cheng Li, Martin Vigoureux, and John Scudder for their comments and review of this document. The authors would like to thank Sue Hares for her detailed shepherd review that helped in improving the document.

9. Contributors

Eric Rosen
Juniper Networks
US

Email: erosen@juniper.net

Arjun Sreekantiah
Cisco Systems
US

Email: asreekan@cisco.com

Acee Lindem
Cisco Systems
US

Email: acee@cisco.com

Siva Sivabalan
Cisco Systems
US

Email: msiva@cisco.com

Imtiyaz Mohammad
Arista Networks
India

Email: imtiyaz@arista.com

Gaurav Dawra
Cisco Systems
US

Email: gdawra.ietf@gmail.com

Peng Shaofu
ZTE Corporation
China

Email: peng.shaofu@zte.com.cn

10. References

10.1. Normative References

- [I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-22 (work in progress), March 2022.
- [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>>.
- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <<https://www.rfc-editor.org/info/rfc3032>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, DOI 10.17487/RFC4360, February 2006, <<https://www.rfc-editor.org/info/rfc4360>>.
- [RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <<https://www.rfc-editor.org/info/rfc4760>>.
- [RFC7606] Chen, E., Ed., Scudder, J., Ed., Mohapatra, P., and K. Patel, "Revised Error Handling for BGP UPDATE Messages", RFC 7606, DOI 10.17487/RFC7606, August 2015, <<https://www.rfc-editor.org/info/rfc7606>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [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>>.

- [RFC8402] Filtsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8660] Bashandy, A., Ed., Filtsfils, C., Ed., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with the MPLS Data Plane", RFC 8660, DOI 10.17487/RFC8660, December 2019, <<https://www.rfc-editor.org/info/rfc8660>>.
- [RFC8664] Sivabalan, S., Filtsfils, C., Tantsura, J., Henderickx, W., and J. Hardwick, "Path Computation Element Communication Protocol (PCEP) Extensions for Segment Routing", RFC 8664, DOI 10.17487/RFC8664, December 2019, <<https://www.rfc-editor.org/info/rfc8664>>.
- [RFC8754] Filtsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [RFC8986] Filtsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.
- [RFC9012] Patel, K., Van de Velde, G., Sangli, S., and J. Scudder, "The BGP Tunnel Encapsulation Attribute", RFC 9012, DOI 10.17487/RFC9012, April 2021, <<https://www.rfc-editor.org/info/rfc9012>>.

10.2. Informational References

- [RFC4272] Murphy, S., "BGP Security Vulnerabilities Analysis", RFC 4272, DOI 10.17487/RFC4272, January 2006, <<https://www.rfc-editor.org/info/rfc4272>>.
- [RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <<https://www.rfc-editor.org/info/rfc4456>>.

[RFC6952] Jethanandani, M., Patel, K., and L. Zheng, "Analysis of BGP, LDP, PCEP, and MSDP Issues According to the Keying and Authentication for Routing Protocols (KARP) Design Guide", RFC 6952, DOI 10.17487/RFC6952, May 2013, <<https://www.rfc-editor.org/info/rfc6952>>.

Authors' Addresses

Stefano Previdi
Huawei Technologies
IT

Email: stefano@previdi.net

Clarence Filsfils
Cisco Systems
Brussels
BE

Email: cfilsfil@cisco.com

Ketan Talaulikar (editor)
Arrcus Inc
India

Email: ketant.ietf@gmail.com

Paul Mattes
Microsoft
One Microsoft Way
Redmond, WA 98052
USA

Email: pamattes@microsoft.com

Dhanendra Jain
Google

Email: ghanendra.ietf@gmail.com

Steven Lin
Google

Email: stevenlin@google.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 26, 2022

Z. Li
L. Li
Huawei
H. Chen
Futurewei
C. Loibl
Next Layer Communications
G. Mishra
Verizon Inc.
Y. Fan
Casa Systems
Y. Zhu
China Telecom
L. Liu
Fujitsu
X. Liu
Volta Networks
August 25, 2021

BGP Flow Specification for SRv6
draft-li-idr-flowspec-srv6-07

Abstract

This document proposes extensions to BGP Flow Specification for SRv6 for filtering packets with a SRv6 SID that matches a sequence of conditions.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD 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.

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 February 26, 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction	2
2. Definitions and Acronyms	4
3. The Flow Specification Encoding for SRv6	4
3.1. Type TBD1 - Some Parts of SID	5
3.2. Encoding Examples	7
3.2.1. Example 1	7
4. Security Considerations	7
5. IANA Considerations	7
6. Acknowledgments	8
7. References	8
7.1. Normative References	8
7.2. Informative References	9
Authors' Addresses	9

1. Introduction

[RFC8955] describes in details about a new BGP NLRI to distribute a flow specification, which is an n-tuple comprising a sequence of matching criteria that can be applied to IP traffic. [RFC8956] extends [RFC8955] to make it also usable and applicable to IPv6 data packets. [I-D.ietf-idr-flowspec-l2vpn] extends the flow-spec rules for layer 2 Ethernet packets. [I-D.hares-idr-flowspec-v2] specifies BGP Flow Specification Version 2.

Segment Routing (SR) for unicast traffic has been proposed to cope with the usecases in traffic engineering, fast re-reroute, service chain, etc. SR architecture can be implemented over an IPv6 data plane using a new type of IPv6 extension header called Segment Routing Header (SRH) [I-D.ietf-6man-segment-routing-header]. SRv6 Network Programming [RFC8986] defines the SRv6 network programming concept and its most basic functions. An SRv6 SID may have the form of LOC:FUNCT:ARG::.

LOC: Each operator is free to use the locator length it chooses. Most often the LOC part of the SID is routable and leads to the node which instantiates that SID.

FUNCT: The FUNCT part of the SID is an opaque identification of a local function bound to the SID. (e.g. End: Endpoint, End.X, End.T, End.DX2 etc.).

ARG: A function may require additional arguments that would be placed immediately after the FUNCT.

This document specifies one new BGP Flow Specification (FS) component type to support Segment Routing over IPv6 data plane (SRv6) filtering for BGP Flow Specification Version 2. The match field is destination address of IPv6 header, but it's a SRv6 SID from SRH rather than a traditional IPv6 address (refer to Figure 1). To support these features, a Flowspec version that is IPv6 capable (i.e., AFI = 2) MUST be used. These match capabilities of the features MAY be permitted to match when there is an accompanying SRH.

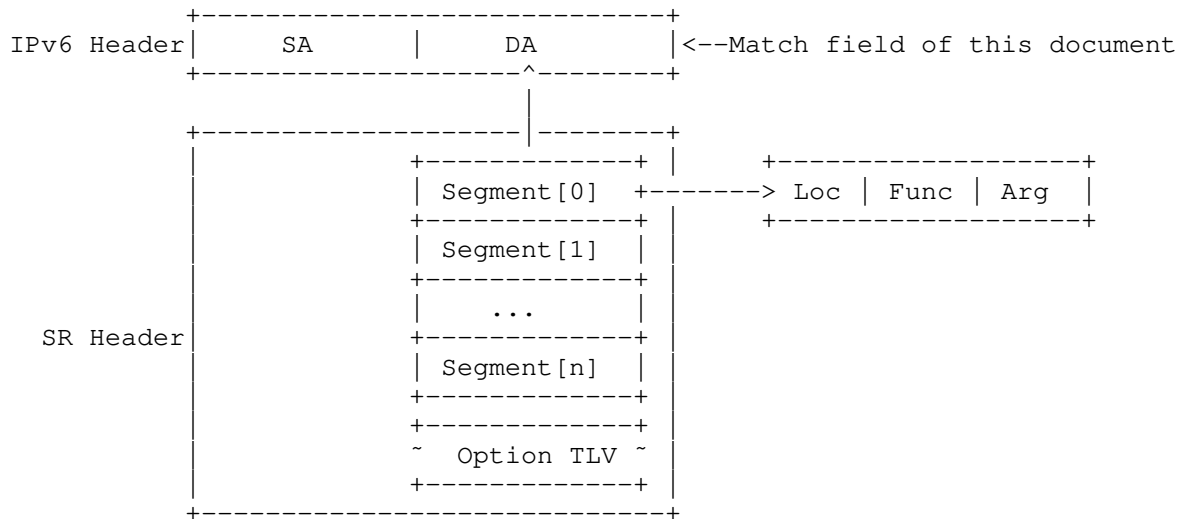


Figure 1: Match Field

2. Definitions and Acronyms

- o FS: Flow Specification
- o BGP-FS: Border Gateway Protocol (BGP) Flow Specification (FS)
- o SR: Segment Routing
- o SRH: SR Header.
- o SRv6: IPv6 Segment Routing, SRv6 is a method of forwarding IPv6 packets on the network based on the concept of source routing.
- o SID: Segment Identifier
- o BSID: Binding SID

3. The Flow Specification Encoding for SRv6

The Flow Specification NLRI-type consists of several optional components, each of which begins with a type field (1 octet) followed by a variable length parameter. 13 component types are defined in [RFC8955] and [RFC8956] for IPv4 and IPv6. This document defines one component type for SRv6.

3.1. Type TBD1 - Some Parts of SID

[RFC8986] defines the format of SID is LOC:FUNCT:ARG::. In some scenarios, traffic packets can just match Locator, Function ID, Arguments or some combinations of these different fields. In order to match a part of SID, its prior parts need to be examined and matched first. For example, in order to match the Function ID (FUNCT), the Locator (LOC) needs to be examined and matched first. The new component type TBD1 defined below is for matching some parts of SID.

Encoding: <type, LOC-Len, FUNCT-Len, ARG-Len, [op, value]+>

- o type (1 octet): This indicates the new component type (TBD1, which is to be assigned by IANA).
- o LOC-Len (1 octet): This indicates the length in bits of LOC in SID.
- o FUNCT-Len (1 octet): This indicates the length in bits of FUNCT in SID.
- o ARG-Len (1 octet): This indicates the length in bits of ARG in SID.
- o [op, value]+: This contains a list of {operator, value} pairs that are used to match some parts of SID.

The total of three lengths (i.e., LOC length + FUNCT length + ARG length) MUST NOT be greater than 128. If it is greater than 128, an error occurs and Error Handling is applied according to [RFC7606] and [RFC4760].

The operator (op) byte is encoded as:

0	1	2	3	4	5	6	7
e	a	field type			lt	gt	eq

where the behavior of each operator bit has clear symmetry with that of [RFC8955]'s Numeric Operator field.

e - end-of-list bit. Set in the last {op, value} pair in the sequence.

a - AND bit. If unset, the previous term is logically ORed with the current one. If set, the operation is a logical AND. It should be

unset in the first operator byte of a sequence. The AND operator has higher priority than OR for the purposes of evaluating logical expressions.

field type:

```

000:  SID's LOC
001:  SID's FUNCT
010:  SID's ARG
011:  SID's LOC:FUNCT
100:  SID's FUNCT:ARG
101:  SID's LOC:FUNCT:ARG

```

For an unknown type, Error Handling is applied according to [RFC7606] and [RFC4760].

lt - less than comparison between data' and value'.

gt - greater than comparison between data' and value'.

eq - equality between data' and value'.

The data' and value' used in lt, gt and eq are indicated by the field type in a operator and the value field following the operator.

The value field depends on the field type and has the value of SID's some parts rounding up to bytes (refer to the table below).

Field Type	Value
SID's LOC	value of LOC bits
SID's FUNCT	value of FUNCT bits
SID's ARG	value of ARG bits
SID's LOC:FUNCT	value of LOC:FUNCT bits
SID's FUNCT:ARG	value of FUNCT:ARG bits
SID's LOC:FUNCT:ARG	value of LOC:FUNCT:ARG bits

3.2. Encoding Examples

3.2.1. Example 1

An example of a Flow Specification NLRI encoding for: all SRv6 packets to LOC 2001:db8:3::/48 and FUNCT {range [0100, 0300]}.

```

      Some Parts of SID
      |
length  v      LOC==20010db80003  FUN>=100  FUN<=300
0x12    0f      01 2001 0db8 0003  4b 0100  bd 0300
           ^   ^   ^
           |   |   |
      Length of LOC  FUN  ARG

```

Decoded:

Value		
0x12	length	18 octets (if len<240, 1 octet)
TBD1(0x0f)	type	type TBD1(0x0f) - Some Parts of SID
0x30	LOC Length	= 48 (bits)
0x10	FUNCT Length	= 16 (bits)
0x40	ARG Length	= 64 (bits)
0x01	op	LOC ==
0x2001	value	LOC's value = 2001:db8:3
0x0db8		
0x0003		
0x4b	op	"AND", FUNCT >=
0x0100	value	FUNCT's value = 0100
0xbd	op	end-of-list, "AND", FUNCT <=
0x0300	value	FUNCT's value = 0300

4. Security Considerations

No new security issues are introduced to the BGP protocol by this specification over the security considerations in [RFC8955] and [RFC8956].

5. IANA Considerations

Under "Flow Spec Component Types" registry, IANA is requested to assign the following values:

Value	IPv4 Name	IPv6 Name	Reference
TBD1	Unassigned	Some Parts of SID	This Document

6. Acknowledgments

The authors would like to thank Joel Halpern, Jeffrey Haas, Ketan Talaulikar, Aijun Wang, Dhruv Dhody, Shunwan Zhuang and Rainsword Wang for their valuable suggestions and comments on this draft.

7. References

7.1. Normative References

- [I-D.hares-idr-flowspec-v2]
Hares, S. and D. Eastlake, "BGP Flow Specification Version 2", draft-hares-idr-flowspec-v2-02 (work in progress), July 2021.
- [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>>.
- [RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <<https://www.rfc-editor.org/info/rfc4760>>.
- [RFC7153] Rosen, E. and Y. Rekhter, "IANA Registries for BGP Extended Communities", RFC 7153, DOI 10.17487/RFC7153, March 2014, <<https://www.rfc-editor.org/info/rfc7153>>.
- [RFC7606] Chen, E., Ed., Scudder, J., Ed., Mohapatra, P., and K. Patel, "Revised Error Handling for BGP UPDATE Messages", RFC 7606, DOI 10.17487/RFC7606, August 2015, <<https://www.rfc-editor.org/info/rfc7606>>.
- [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>>.
- [RFC8955] Loibl, C., Hares, S., Raszuk, R., McPherson, D., and M. Bacher, "Dissemination of Flow Specification Rules", RFC 8955, DOI 10.17487/RFC8955, December 2020, <<https://www.rfc-editor.org/info/rfc8955>>.
- [RFC8956] Loibl, C., Ed., Raszuk, R., Ed., and S. Hares, Ed., "Dissemination of Flow Specification Rules for IPv6", RFC 8956, DOI 10.17487/RFC8956, December 2020, <<https://www.rfc-editor.org/info/rfc8956>>.

7.2. Informative References

- [I-D.ietf-6man-segment-routing-header]
Filsfils, C., Dukes, D., Previdi, S., Leddy, J.,
Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header
(SRH)", draft-ietf-6man-segment-routing-header-26 (work in
progress), October 2019.
- [I-D.ietf-idr-flowspec-l2vpn]
Hao, W., Eastlake, D. E., Litkowski, S., and S. Zhuang,
"BGP Dissemination of L2 Flow Specification Rules", draft-
ietf-idr-flowspec-l2vpn-17 (work in progress), May 2021.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer,
D., Matsushima, S., and Z. Li, "Segment Routing over IPv6
(SRv6) Network Programming", RFC 8986,
DOI 10.17487/RFC8986, February 2021,
<<https://www.rfc-editor.org/info/rfc8986>>.

Authors' Addresses

Zhenbin Li
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China

Email: lizhenbin@huawei.com

Lei Li
Huawei
156 Beiqing Road
Beijing 100095
P.R. China

Email: lily.lilei@huawei.com

Huaimo Chen
Futurewei
Boston, MA
USA

Email: Huaimo.chen@futurewei.com

Christoph Loibl
Next Layer Communications
Mariahilfer Guertel 37/7
Vienna 1150
AT

Email: cl@tix.at

Gyan S. Mishra
Verizon Inc.
13101 Columbia Pike
Silver Spring MD 20904
USA

Phone: 301 502-1347
Email: gyan.s.mishra@verizon.com

Yanhe Fan
Casa Systems
USA

Email: yfan@casa-systems.com

Yongqing Zhu
China Telecom
109, West Zhongshan Road, Tianhe District
Guangzhou 510000
China

Email: zhuyq8@chinatelecom.cn

Lei Liu
Fujitsu
USA

Email: liulei.kddi@gmail.com

Xufeng Liu
Volta Networks
McLean, VA
USA

Email: xufeng.liu.ietf@gmail.com

Inter-Domain Routing
Internet-Draft
Updates: 4271 (if approved)
Intended status: Standards Track
Expires: March 26, 2020

J. Snijders
NTT
M. Aelmans
Juniper Networks
September 23, 2019

Revised BGP Maximum Prefix Limits
draft-sa-idr-maxprefix-00

Abstract

This document updates RFC4271 by revising control mechanism which limit the negative impact of route leaks (RFC7908) and/or resource exhaustion in Border Gateway Protocol (BGP) implementations.

Requirements Language

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

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 March 26, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction	2
2. Changes to RFC4271 Section 6	2
3. Changes to RFC4271 Section 8	3
4. BGP Yang Model Considerations – PERHAPS REMOVE BEFORE PUBLICATION	4
5. Changes to RFC4271 Section 9	4
6. Security Considerations	6
7. IANA Considerations	6
8. Acknowledgments	6
9. Implementation status – RFC EDITOR: REMOVE BEFORE PUBLICATION	6
10. Appendix: Implementation Guidance	7
11. References	8
11.1. Normative References	8
11.2. Informative References	8
Authors' Addresses	8

1. Introduction

This document updates [RFC4271] by revising control mechanism which limit the negative impact of route leaks [RFC7908] and/or resource exhaustion in Border Gateway Protocol (BGP) implementations. While [RFC4271] described methods to tear down BGP sessions or discard UPDATES after certain thresholds are exceeded, some nuances in this specification were missing resulting in inconsistencies between BGP implementations. In addition to clarifying "inbound maximum prefix limits", this document also introduces a specification for "outbound maximum prefix limits".

2. Changes to RFC4271 Section 6

This section updates [RFC4271] to specify what events can result in AutomaticStop (Event 8) in the BGP FSM.

The following paragraph replaces the second paragraph of Section 6.7 (Cease), which starts with "A BGP speaker MAY support" and ends with "The speaker MAY also log this locally.":

A BGP speaker MAY support the ability to impose a locally-configured, upper bound on the number of address prefixes the speaker is willing to accept from a neighbor (inbound maximum prefix limit) or send to a neighbor (outbound prefix limit). The limit on the prefixes accepted from a neighbor can be applied before policy processing (Pre-Policy) or after policy processing (Post-Policy). Outbound prefix limits MUST be measured after policy since the Policy (even a policy of "send all") is run before determining what can be sent. When the upper bound is reached, the speaker, under control of local configuration, either:

- A. Discards new address prefixes to or from the neighbor (while maintaining the BGP connection with the neighbor)
- B. Terminates the BGP connection with the neighbor

If the BGP peer uses option (b) where the limit causes a CEASE Notification, then the CEASE error codes should use:

Subcode	Symbolic Name
1	Maximum Number of Prefixes Reached
TBD	Threshold exceeded: Self-Destructing, Maximum Number of Prefixes Send

The speaker MAY also log this locally.

3. Changes to RFC4271 Section 8

This section updates Section 8 [RFC4271], the paragraph that starts with "One reason for an AutomaticStop event is" and ends with "The local system automatically disconnects the peer." is replaced with:

Possible reasons for an AutomaticStop event are: A BGP speaker receives an UPDATE messages with a number of prefixes for a given peer such that the total prefixes received exceeds the maximum number of prefixes configured (either "Pre-Policy" or "Post-Policy"), or announces more prefixes than through local configuration allowed to. The local system automatically disconnects the peer.

4. BGP Yang Model Considerations - PERHAPS REMOVE BEFORE PUBLICATION

In [I-D.ietf-idr-bgp-model] in container 'prefix-limit', a leaf named "max-prefixes" exists. The authors recommend the BGP Yang Model to be revised to contain the following leaves:

max-prefixes-inbound-pre-policy

max-prefixes-inbound-post-policy

max-prefixes-outbound

In addition to the above, the authors suggest that the BGP Yang Model is extended in such a way that per peer per AFI/SAFI pair an operator can specify whether to tear down the session or discard sending or receiving updates.

5. Changes to RFC4271 Section 9

This section updates [RFC4271] by adding a subsection after Section 9.4 (Originating BGP routes) to specify various events that can lead up to AutomaticStop (Event 8) in the BGP FSM.

9.5 Maximum Prefix Limits

9.5.1 Pre-Policy Inbound Maximum Prefix Limits

The Adj-RIBs-In stores routing information learned from inbound UPDATE messages that were received from another BGP speaker Section 3.2 [RFC4271]. The pre-policy limit uses the number of NLRIs per Address Family Identifier (AFI) per Subsequent Address Family Identifier (SAFI) as input into its threshold comparisons. For example, when an operator configures the pre-policy limit for IPv4 Unicast to be 50 on a given EBGP session, and the other BGP speaker announces its 51st IPv4 Unicast NLRI, the session MUST be terminated.

Pre-policy limits are particularly useful to help dampen the effects of full table route leaks and memory exhaustion when the implementation stores rejected routes.

9.5.2 Post-Policy Inbound Maximum Prefix Limits

RFC4271 describes a Policy Information Base (PIB) that contains local policies that can be applied to the information in the Routing Information Base (RIB). The post-policy limit uses the number of NLRIs per Address Family Identifier (AFI) per Subsequent Address Family Identifier (SAFI), after application of the Import Policy as input into its threshold comparisons. For example, when an operator configures the post-policy limit for IPv4 Unicast to be 50 on a given EBGp session, and the other BGP speaker announces a hundred IPv4 Unicast routes of which none are accepted as a result of the local import policy (and thus not considered for the Loc-RIB by the local BGP speaker), the session is not terminated.

Post-policy limits are useful to help prevent FIB exhaustion and prevent accidental BGP session teardown due to prefixes not accepted by policy anyway.

9.5.3 Outbound Maximum Prefix Limits

An operator MAY configure a BGP speaker to terminate its BGP session with a neighbor when the number of address prefixes to be advertised to that neighbor exceeds a locally configured post-policy upper limit. The BGP speaker then MUST send the neighbor a NOTIFICATION message with the Error Code Cease and the Error Subcode "Threshold reached: Maximum Number of Prefixes Send". Implementations MAY support additional actions. The Hard Cease action is defined in [RFC8538].

Reporting when thresholds have been exceeded is an implementation specific consideration, but SHOULD include methods such as Syslog [RFC5424]. By definition, Outbound Maximum Prefix Limits are Post-Policy.

The Adj-RIBs-Out stores information selected by the local BGP speaker for advertisement to its neighbors. The routing information stored in the Adj-RIBs-Out will be carried in the local BGP speaker's UPDATE messages and advertised to its neighbors Section 3.2 [RFC4271]. The Outbound Maximum Prefix Limit uses the number of NLRIs per Address Family Identifier (AFI) per Subsequent Address Family Identifier (SAFI), after application of the Export Policy, as input into its threshold comparisons. For example, when an operator configures the Outbound Maximum Prefix Limit for IPv4 Unicast to be 50 on a given EBGp session, and were about to announce its 51st IPv4 Unicast NLRI to the other BGP speaker as a result of the local export policy, the session MUST be terminated.

Outbound Maximum Prefix Limits are useful to help dampen the negative effects of a misconfiguration in local policy. In many cases, it would be more desirable to tear down a BGP session rather than causing or propagating a route leak.

6. Security Considerations

Maximum Prefix Limits are an essential tool for routing operations and SHOULD be used to increase stability.

7. IANA Considerations

This memo requests that IANA assigns a new subcode named "Threshold exceeded: Self-Destructing, Maximum Number of Prefixes Send" in the "Cease NOTIFICATION message subcodes" registry under the "Border Gateway Protocol (BGP) Parameters" group.

8. Acknowledgments

The authors would like to thank Saku Ytti and John Heasley (NTT), Jeff Haas, Colby Barth and John Scudder (Juniper Networks), Martijn Schmidt (i3D.net), Teun Vink (BIT), Sabri Berisha (eBay), Martin Pels (Quanza), Steven Bakker (AMS-IX), Aftab Siddiqui (ISOC), Yu Tianpeng, Ruediger Volk (Deutsche Telekom), Robert Raszuk (Bloomberg), Jakob Heitz (Cisco), and Susan Hares (Hickory Hill Consulting) for their support, insightful review, and comments.

9. Implementation status - RFC EDITOR: REMOVE 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.

The below table provides an overview (as of the moment of writing) of which vendors have produced implementation of inbound or outbound maximum prefix limits. Each table cell shows the applicable configuration keywords if the vendor implemented the feature.

Vendor	Inbound Pre-Policy	Inbound Post-Policy	Outbound
Cisco IOS XR		maximum-prefix	
Cisco IOS XE		maximum-prefix	
Juniper Junos OS	prefix-limit	accepted-prefix-limit, or prefix-limit combined with 'keep none'	
Nokia SR OS	prefix-limit		
NIC.CZ BIRD	'import keep filtered' combined with 'receive limit'	'import limit' or 'receive limit'	export limit
OpenBSD OpenBGPD	max-prefix		
Arista EOS	maximum-routes	maximum-accepted-routes	
Huawei VRPv5	peer route-limit		
Huawei VRPv8	peer route-limit	peer route-limit accept-prefix	

First presented by Snijders at [RIPE77]

Table 1: Maximum prefix limits capabilities per implementation

10. Appendix: Implementation Guidance

1) make it clear who does what: if A sends too many prefixes to B A should see "ABC" in log B should see "DEF" in log to make it clear which of the two parties does what 2) recommended by default automatically restart after between 15 and 30 minutes

11. References

11.1. Normative References

- [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>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [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>>.
- [RFC8538] Patel, K., Fernando, R., Scudder, J., and J. Haas, "Notification Message Support for BGP Graceful Restart", RFC 8538, DOI 10.17487/RFC8538, March 2019, <<https://www.rfc-editor.org/info/rfc8538>>.

11.2. Informative References

- [I-D.ietf-idr-bgp-model] Jethanandani, M., Patel, K., and S. Hares, "BGP YANG Model for Service Provider Networks", draft-ietf-idr-bgp-model-06 (work in progress), June 2019.
- [RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, DOI 10.17487/RFC5424, March 2009, <<https://www.rfc-editor.org/info/rfc5424>>.
- [RFC7908] Sriram, K., Montgomery, D., McPherson, D., Osterweil, E., and B. Dickson, "Problem Definition and Classification of BGP Route Leaks", RFC 7908, DOI 10.17487/RFC7908, June 2016, <<https://www.rfc-editor.org/info/rfc7908>>.
- [RIPE77] Snijders, J., "Robust Routing Policy Architecture", May 2018, <https://ripe77.ripe.net/wp-content/uploads/presentations/59-RIPE77_Snijders_Routing_Policy_Architecture.pdf>.

Authors' Addresses

Job Snijders
NTT
Theodorus Majofskistraat 100
Amsterdam 1065 SZ
The Netherlands

Email: job@ntt.net

Melchior Aelmans
Juniper Networks
Boeing Avenue 240
Schiphol-Rijk 1119 PZ
The Netherlands

Email: maelmans@juniper.net

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: August 6, 2020

Yimin Shen
Juniper Networks
Ravi Singh
Individual Contributor
Yuji Kamite
NTT Communications
February 3, 2020

BGP Flexible Color-Based Tunnel Selection
draft-shen-idr-flexible-color-tunnel-selection-01

Abstract

This document discusses color-based tunnel selection for BGP payload prefixes. It defines a set of extended mapping modes, and describes how to use them to construct tunnel selection schemes for flexible tunnel selection. Tunnel selection schemes can be implemented as policies on routers performing tunnel selection, or signaled by next hop routers or a central controller by using the BGP extensions specified in this document.

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 August 6, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction	2
2. Specification of Requirements	4
3. Extended Mapping Modes	4
4. Tunnel Selection Scheme and Operation	6
5. Provisioning of Tunnel Selection Schemes	7
6. BGP Tunnel Encapsulation Attribute Extensions	8
6.1. Wildcard Tunnel Type	8
6.2. Color Tunnel Selection Scheme Sub-TLV	8
6.2.1. Extended Mapping Mode Sub-sub-TLV	9
6.2.2. Encoding	10
6.2.3. Decoding	10
6.3. Association between Color Tunnel Selection Scheme Sub-TLV and Tunnel Type	11
7. Relationship with Color-Only Bits of Color Extended Community	11
8. IANA Considerations	11
9. Security Considerations	12
10. Acknowledgements	12
11. References	12
11.1. Normative References	12
11.2. Informative References	13
Authors' Addresses	13

1. Introduction

In an overlay network using BGP for payload prefix distribution, transporting the packets of a payload prefix from a BGP router to the next BGP router relies on the selection of a transport tunnel. This selection may be based on various attributes of the prefix, such as BGP next hop, color, and information in the Tunnel Encapsulation Attribute [BGP-TUNNEL-ENCAP], etc.

In one tunnel selection model, color is used as a primary criterion along with BGP next hop or tunnel endpoint address (contained in a Tunnel Endpoint sub-TLV of the Tunnel Encapsulation Attribute). This model is referred to as "color-based" tunnel selection in this document. The model is gaining many use cases today due to its general applicability. In particular, color as a generic notion may be used to represent a broad range of network attributes, such as virtual topology, network slice, path computation algorithm, TE constraint, administrative profile, etc. For some of the attributes,

there may not be a convenient mechanism to associate them with payload prefixes or tunnels, or to distribute them in a network, especially across domains or to a central controller. When these attributes need to be considered in tunnel selection, mapping them to colors and performing color-based tunnel selection will provide a generic solution.

The procedures in this document is relevant to color-based tunnel selection. In general, payload prefixes may be associated with colors through configuration or a Color Extended Community [RFC5512]. Transport tunnels may also be associated with colors through configuration (e.g. RSVP and LDP tunnels), a Color Extended Community (e.g. BGP LU), or a color embedded in BGP NLRI (e.g. BGP SR-TE policy [BGP-SR-POLICY]), etc. These payload prefixes and tunnels are called "colored payload prefixes" and "colored tunnels", respectively.

Normally, a payload prefix of color X is intended to be mapped to a tunnel of the same color X. This is considered as the default mapping mode of color-based tunnel selection. In some cases, when a tunnel of color X cannot be found or established, or when a previously mapped tunnel of color X fails, a network operator may want to map the payload prefix by attempting other modes, e.g. selecting a tunnel of another color Y, a tunnel without a color, a tunnel of color X but with an IPv4-mapped IPv6 endpoint address, and so on. Note that the colors may represent network slices, virtual topologies, path computation algorithms, etc. Hence, these modes will provide the flexibility and enable the operator to take the full transport capability of the network. In this document, these modes are called "extended mapping modes", and the procedure of automatically executing them in a user-defined order is called "fallback".

This document defines a set of extended mapping modes to complement the default mapping mode. It introduces the notion of "tunnel selection scheme". A tunnel selection scheme is an ordered list of extended mapping modes to be automatically executed in tunnel selection. When a tunnel is not selected by using the first mode in the list, fallback is performed by attempting the second mode, the third mode, and so on, until a tunnel is selected or the list is exhausted.

Color-based tunnel selection for uncolored payload prefixes is also considered in this document as a special case. By using a tunnel selection scheme, a colored or uncolored tunnel may be selected for an uncolored payload prefix in a flexible manner.

2. Specification of Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174].

3. Extended Mapping Modes

This document defines a set of extended mapping modes for flexible color-based tunnel selection. Each mode specifies how a payload prefix's endpoint IP address (derived from BGP next hop or a Tunnel Endpoint sub-TLV in the Tunnel Encapsulation Attribute [BGP-TUNNEL-ENCAP]) and color are used to select a tunnel. The document assumes that each payload prefix SHOULD have a single color for tunnel selection purpose or no color, and each tunnel SHOULD have a single color or no color.

In the definitions of the extended mapping modes below, N represents a payload prefix's endpoint IP address, and C represents its color. An uncolored payload prefix does not have a color. An extended mapping mode may have multiple steps for sub-level fallback within it. The steps are executed sequentially. The mode is completed as soon as a tunnel is successfully selected in one of the steps, and the rest steps are skipped.

(1) IP-color, optionally with a fallback color list of {C1, ..., Cn}

- If the payload prefix has a color C, select a tunnel with endpoint address N and color C.
- Select a tunnel with endpoint address N and color C1.
- ...
- Select a tunnel with endpoint address N and color Cn.

(2) Color-only, optionally with a fallback color list of {C1, ..., Cn}

- If the payload prefix has a color C, select a tunnel with color C, regardless of the tunnel's endpoint address.
- Select a tunnel with color C1, regardless of tunnel's endpoint address.
- ...

- Select a tunnel with color Cn, regardless of tunnel's endpoint address.

(3) IP-any-color

- Select a tunnel with endpoint address N and any color.

(4) IP-only

- Select a tunnel with endpoint address N and without a color.

(5) Converted-IPv6

This mode is applicable when N is an IPv4 address. Assume N' is the IPv6 address mapped from N.

- Select a tunnel with endpoint address N' and without a color.

(6) Converted-IPv6-color, optionally a fallback color list of {C1, ..., Cn}

This mode is applicable when N is an IPv4 address. Assume N' is the IPv6 address mapped from N.

- If the payload prefix has a color C, select a tunnel with endpoint address N' and color C.
- Select a tunnel with endpoint address N' and color C1.
- ...
- Select a tunnel with endpoint address N' and color Cn.

(7) Converted-IPv6-any-color

This mode is applicable when N is an IPv4 address. Assume N' is the IPv6 address mapped from N.

- Select a tunnel with endpoint address N' and any color.

(8) Color-profile

- If the payload prefix has a color C, use C as key to look up a profile to construct tunnel selection criteria and select a tunnel.

As shown above, the IP-color, Color-only, and Converted-IPv6-color modes may have a fallback color list for sub-level fallback across the colors.

This list is not exhaustive. More modes MAY be defined in the future.

4. Tunnel Selection Scheme and Operation

A tunnel selection scheme is defined as an ordered list of extended mapping modes (Section 3) to be automatically executed in tunnel selection. The first mode in the list is called a "primary" mode, and all the subsequent modes are called "fallback" modes. A scheme MUST have a primary mode, but MAY or MAY not have any fallback mode.

When a scheme is executed for a payload prefix, the modes in the list are executed sequentially, and within each mode, the steps of sub-level fallback are executed sequentially. When a tunnel is selected in a particular step in a particular mode, the scheme is completed, and all subsequent steps of the mode and all the subsequent modes in the list are skipped. If no tunnel is selected when the list is exhausted, the payload prefix will remain in unresolved state for transport.

In the case where a previously selected tunnel becomes inoperative, the scheme SHOULD be run to reselect a tunnel. In the case where a tunnel was previously selected and later another tunnel of higher preference (in the tunnel selection scheme or in a fallback color list) becomes available, the new tunnel MAY be selected to replace the current tunnel. This procedure is called a reversion. It may be performed manually by a network operator, or triggered automatically by the situation.

The following are some examples of tunnel selection schemes.

Example 1:

A payload prefix has a tunnel endpoint IPv4 address 203.0.113.1 and a color RED. It is associated with the following tunnel selection scheme:

- (1) IP-color
- (2) Converted-IPv6-color
- (3) IP-only

The intended tunnel selection procedure is:

(1) Find a tunnel with endpoint IPv4 address 203.0.113.1 and color RED.

(2) If the above is unsuccessful, convert the IPv4 address to an IPv6 address 2002:cb00:7101::/64. Find a tunnel with endpoint IPv6 address 2002:cb00:7101::/64 and color RED.

(3) If the above is unsuccessful, find a tunnel with endpoint IPv4 address 203.0.113.1 and without a color.

Example 2:

A prefix has a tunnel endpoint IPv4 address 203.0.113.1 and a color RED. It is associated with the following tunnel selection scheme:

- (1) IP-color, with a fallback color list = {BLUE, GREEN}
- (2) Converted-IPv6-color, with a fallback color list = {WHITE}
- (3) IP-only

The intended tunnel selection procedure is:

(1) Find a tunnel with endpoint IPv4 address 203.0.113.1 and color RED. If it is unsuccessful, find a tunnel with endpoint IPv4 address 203.0.113.1 and color BLUE. If it is unsuccessful, find a tunnel with endpoint IPv4 address 203.0.113.1 and color GREEN.

(2) If the above is unsuccessful, convert the IPv4 address to an IPv6 address 2002:cb00:7101::/64. Find a tunnel with endpoint IPv6 address 2002:cb00:7101::/64 and color RED. If it is unsuccessful, find a tunnel with endpoint IPv6 address 2002:cb00:7101::/64 and color WHITE.

(3) If the above is unsuccessful, find a tunnel with endpoint IPv4 address 203.0.113.1 and without a color.

5. Provisioning of Tunnel Selection Schemes

A tunnel selection scheme MAY be provisioned for a payload prefix on a router which performs tunnel selection. In this case, the scheme may be implemented as a policy on the router. The configuration of such policy varies by vendors, and hence is out of the scope of this document.

A tunnel selection scheme MAY also be provisioned on a router or a central controller which originates the UPDATE message of a payload prefix, and then distributed to a router(s) which will perform tunnel

selection. To facilitate this, the document introduces a new "Color Tunnel Selection Scheme" sub-TLV (Section 6) to the Tunnel Encapsulation Attribute to carry the information. As color-based tunnel selection is typically across all tunnel types, the document also introduces a new "Wildcard" tunnel type to the Tunnel Encapsulation Attribute. When the tunnel selection scheme contained in a Color Tunnel Selection Scheme sub-TLV is applicable to all tunnel types, the top-level Tunnel Encapsulation Attribute TLV SHOULD set tunnel type to Wildcard.

In the case where a payload prefix has one scheme configured as a policy on a router, and another scheme received in a Color Tunnel Selection Scheme sub-TLV, the router SHOULD treat the policy in preference to the received information.

If a payload prefix does not have a tunnel selection scheme, the default mapping mode applicable to colored or non-colored payload prefixes SHOULD be used accordingly.

6. BGP Tunnel Encapsulation Attribute Extensions

This section specifies a new "Wildcard" tunnel type and a new Color Tunnel Selection Scheme sub-TLV for the Tunnel Encapsulation Attribute.

6.1. Wildcard Tunnel Type

The Wildcard tunnel type has the semantics of "any tunnel types". It allows a Tunnel Encapsulation Attribute TLV to carry information which is regardless of tunnel type or applicable to all tunnel types. In this document, it is used when a Tunnel Encapsulation Attribute TLV contains a Color Tunnel Selection Scheme sub-TLV which is applicable to all tunnel types.

6.2. Color Tunnel Selection Scheme Sub-TLV

The Color Tunnel Selection Scheme sub-TLV is used to carry the information of a tunnel selection scheme. The sub-TLV is contained in a Tunnel Encapsulation Attribute TLV. If the Tunnel Encapsulation Attribute TLV's tunnel type is Wildcard, the tunnel selection scheme is regardless of tunnel type. If the Tunnel Encapsulation Attribute TLV's tunnel type is a specific type, the tunnel selection scheme is applicable to tunnels of that type. In any case, a Tunnel Encapsulation Attribute TLV MUST not contain more than one Color Tunnel Selection Scheme sub-TLV.

The sub-TLV's Type is TBD (to be allocated by IANA). The sub-TLV's Length is the number of the octets of the sub-TLV's Value field. The

sub-TLV's Value field is composed of one or multiple Extended Mapping Mode sub-sub-TLVs.

6.2.1. Extended Mapping Mode Sub-sub-TLV

An Extended Mapping Mode sub-sub-TLV contains the information of an extended mapping mode. Its encoding is shown in Figure 1.

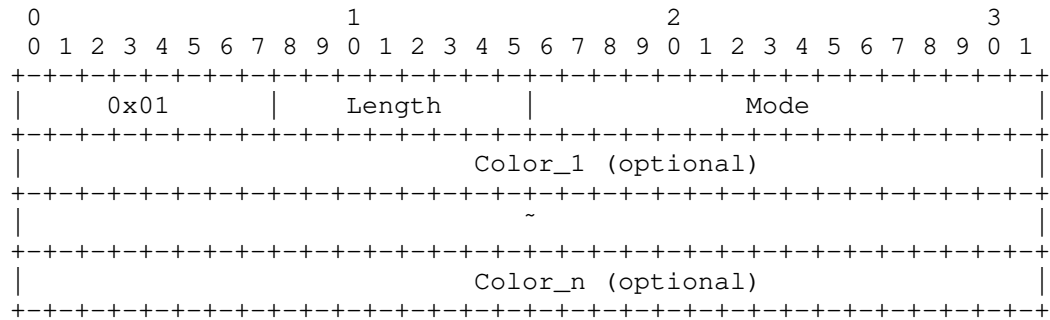


Figure 1

The Extended Mapping Mode sub-sub-TLV's Type is 0x01.

The Extended Mapping Mode sub-sub-TLV's Length is the total number of octets of the sub-sub-TLV's Value field.

The Extended Mapping Mode sub-sub-TLV's Value field contains a 2-octet extended mapping mode defined as below, and an optional fallback color list.

- 1 - IP-color
- 2 - Color-only
- 3 - IP-any-color
- 4 - IP-only
- 5 - Converted-IPv6
- 6 - Converted-IPv6-color
- 7 - Converted-IPv6-any-color
- 8 - Color-profile

The IP-color, Color-only and Converted-IPv6-color modes MAY have an optional fallback color list. The list contains one or multiple 4-octet color values, i.e. Color_1, ..., Color_n, in the order from the highest preference to the lowest preference.

6.2.2. Encoding

Given a tunnel selection scheme, a Color Tunnel Selection Scheme sub-TLV is constructed in the following manner:

- o First, an Extended Mapping Mode sub-sub-TLV containing the primary mode is added. If this mode is IP-Color, Color-Only, or Converted-IPv6-Color, and if cross-color fallback is applicable to this mode, a fallback color list is added to the sub-sub-TLV.
- o If there is one or multiple desired fallback modes, an Extended Mapping Mode sub-sub-TLV containing the first fallback mode is added. If this mode is IP-Color, Color-Only, or Converted-IPv6-Color, and if cross-color fallback is applicable to this mode, a fallback color list is added to the sub-sub-TLV.
- o This process continues, until an Extended Mapping Mode sub-sub-TLV containing the last fallback mode is added. If this mode is IP-Color, Color-Only, or Converted-IPv6-Color, and if cross-color fallback is applicable to this mode, a fallback color list is added to the sub-sub-TLV.

6.2.3. Decoding

When decoding a Color Tunnel Selection Scheme sub-TLV, a receiving router MUST interpret the preference of the contained Extended Mapping Mode sub-sub-TLVs as the order in which they are encoded. If an Extended Mapping Mode sub-sub-TLV contains a mode which is not IP-Color, Color-Only, or Converted-IPv6-Color but has a fallback color list, the entire Color Tunnel Selection Scheme sub-TLV SHOULD be considered as malformed and ignored.

A receiving router MUST consider a payload prefix as having a modified tunnel selection scheme in any of the following situations, and perform tunnel selection accordingly:

- o The payload prefix did not have a valid Color Tunnel Selection Scheme sub-TLV in the previous UPDATE message, and it has one in the latest UPDATE message. Tunnel selection MUST be performed based on the latest tunnel selection scheme.
- o The payload prefix had a valid Color Tunnel Selection Scheme sub-TLV in the previous UPDATE message, but it does not have one in

the latest UPDATE message. Tunnel selection MUST revert to the default color or non-color mapping mode.

- o The payload prefix had a valid Color Tunnel Selection Scheme sub-TLV in the previous UPDATE message, and it has one with different content in the latest UPDATE message. Tunnel selection MUST be performed based on the latest tunnel selection scheme.

6.3. Association between Color Tunnel Selection Scheme Sub-TLV and Tunnel Type

A Color Tunnel Selection Scheme sub-TLV MAY be contained in a Tunnel Encapsulation Attribute TLV of Wildcard tunnel type, indicating that the scheme SHOULD be performed regardless of tunnel type. The sub-TLV MAY also be contained in a Tunnel Encapsulation Attribute TLV of a specific tunnel type, indicating that the scheme SHOULD consider only the tunnels of that type. In the case where a Tunnel Encapsulation Attribute contains a TLV of Wildcard tunnel type and another TLV of a specific tunnel type, and both TLVs contain a Color Tunnel Selection Scheme sub-TLV, tunnel selection for that specific tunnel type SHOULD be based on the corresponding Color Tunnel Selection Scheme sub-TLV.

7. Relationship with Color-Only Bits of Color Extended Community

[RFC8402] and [BGP-SR-POLICY] define two "Color-Only" bits (i.e. CO bits) in the BGP Color Extended Community for color-based tunnel selection in the context of segment routing. Each of the four combinations of the CO bits corresponds to a predefined fallback scheme.

This document complements these documents by supporting more generic and flexible fallback schemes which are user definable. In fact, the predefined fallback schemes of the CO bits can be fully supported by using the Color Tunnel Selection Scheme sub-TLV. When a router advertises an UPDATE message, it SHOULD NOT use a Color Extended Community with CO bits and a Color Tunnel Selection Scheme sub-TLV at the same time, in order to avoid collision between them. If a router receives an UPDATE message containing both objects, it SHOULD give preference to CO bits, and ignore the other. If the tunnel selection scheme is implemented as a policy on the receiving router, the router SHOULD give the preference to the policy.

8. IANA Considerations

This document requires the IANA to allocate a value for the Wildcard tunnel type as a new BGP Tunnel Encapsulation Attribute Type, and a type value for the new Color Tunnel Selection Scheme sub-TLV.

9. Security Considerations

This document introduces procedures for color-based tunnel selection to use tunnel selection schemes. The procedures can cause traffic to be diverted from default path(s). This requires measures to be taken at a number of levels to avoid undesirable transport behaviors and security risks. First, network coloring (i.e. color assignment to network resources, attributes, payload prefixes, tunnels, etc.) MUST be carefully planned and validated at a global level to avoid errors and collisions. Second, tunnel selection schemes MUST be legitimate and always select valid tunnels leading to desired endpoints. For schemes implemented as policies, this SHOULD be ensured by policy configuration. For schemes distributed via Color Tunnel Selection Scheme sub-TLV of BGP Tunnel Encapsulation Attribute, receivers SHOULD use BGP security procedures to validate each originator's identity and detect unauthorized content modification during distribution.

10. Acknowledgements

This document leverages work done by Junan Chen. Thanks to Jeff Hass and Srihari Sangli for their kind reviews and comments which helped to improve the clarity of this document.

11. References

11.1. Normative References

- [RFC5512] Mohapatra, P. and E. Rosen, "The BGP Encapsulation Subsequent Address Family Identifier (SAFI) and the BGP Tunnel Encapsulation Attribute", RFC 5512, DOI 10.17487/RFC5512, April 2009, <<https://www.rfc-editor.org/info/rfc5512>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [BGP-SR-POLICY] Previdi, S., Filsfils, C., Mattes, P., Rosen, E., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-previdi-idr-segment-routing-te-policy (work in progress), 2019.

[BGP-TUNNEL-ENCAP]

Patel, K., Velde, G., and S. Sangli, "The BGP Tunnel Encapsulation Attribute", draft-vandeveld-idr-remote-next-hop (work in progress), 2019.

11.2. Informative References

- [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>>.
- [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>>.

Authors' Addresses

Yimin Shen
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Phone: +1 9785890722
Email: yshen@juniper.net

Ravi Singh
Individual Contributor

Email: ravi.singh.ietf@gmail.com

Yuji Kamite
NTT Communications

Email: y.kamite@ntt.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 16, 2022

H. Wang
Huawei
A. Wang
China Telecom
S. Zhuang
Huawei
February 12, 2022

Destination-IP-Origin-AS Filter for BGP Flow Specification
draft-wang-idr-flowspec-dip-origin-as-filter-05

Abstract

BGP Flowspec mechanism (BGP-FS) [RFC8955] [RFC8956] propagates both traffic Flow Specifications and Traffic Filtering Actions by making use of the BGP NLRI and the BGP Extended Community encoding formats. This document specifies a new BGP-FS component type to support AS-level filtering. The match field is the origin AS number of the destination IP address that is encoded in the Flowspec NLRI. This function is applied in a single administrative domain.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

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 August 16, 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must 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.

Table of Contents

1. Introduction	2
2. Definitions and Acronyms	3
3. The Flow Specification Encoding for Destination-IP-Origin-AS Filter	3
4. Use Case	4
5. Security Considerations	6
6. IANA	6
7. Contributors	6
8. Acknowledgments	6
9. References	7
Authors' Addresses	7

1. Introduction

BGP Flow Specification (BGP-FS) [RFC8955] [RFC8956] defines a new BGP NLRI to distribute traffic flow specification rules via BGP ([RFC4271]). BGP-FS policies have a match condition that may be n-tuple match in a policy, and an action that modifies the packet and forwards/drops the packet. Via BGP, new filter rules can be sent to all BGP peers simultaneously without changing router configuration, and the BGP peer can install these routes in the forwarding table. BGP-FS defines Network Layer Reachability Information (NLRI) format used to distribute traffic flow specification rules. NLRI (AFI=1, SAFI=133) is for IPv4 unicast filtering. NLRI (AFI=1, SAFI=134) is for BGP/MPLS VPN filtering. [I-D.ietf-idr-flowspec-l2vpn][I-D.ietf-idr-flowspec-l2vpn] extends the flow-spec rules for layer 2 Ethernet packets.

This document specifies a new BGP-FS component type to support AS-level filtering. The match field is the origin AS number of the

destination IP address that is encoded in the Flowspec NLRI. This function is applied in a single administrative domain.

2. Definitions and Acronyms

- o FS: Flow Specification
- o Destination-IP-Origin-AS: The origin AS number of the destination IP address

3. The Flow Specification Encoding for Destination-IP-Origin-AS Filter

This document proposes a new flow specification component type that is encoded in the BGP Flowspec NLRI. The following new component type is defined.

- o Destination-IP-Origin-AS

Type TBD1 - Destination-IP-Origin-AS

Encoding: <type (1 octet), [op, value]+>

Contains a set of {operator, value} pairs that are used to match the Destination-IP-Origin-AS (i.e. the origin AS number of the destination IP address).

The operator byte is encoded as:

0	1	2	3	4	5	6	7
+	+	+	+	+	+	+	+
e	a	len	0	lt	gt	eq	
+	+	+	+	+	+	+	+

Where:

e - end-of-list bit. Set in the last {op, value} pair in the list.

a - AND bit. If unset, the previous term is logically ORed with the current one. If set, the operation is a logical AND. It MUST be unset in the Destination-IP-Origin-AS filter.

len - The length of the value field for this operator given as (1 << len). This encodes 1 (len=00), 2 (len=01), 4 (len=10), and 8 (len=11) octets.

lt - less than comparison between data and value.

gt - greater than comparison between data and value.

eq - equality between data and value.

The bits lt, gt, and eq can be combined to produce match the Destination-IP-Origin-AS filter or a range of Destination-IP-Origin-AS filter(e.g. less than AS1 and greater than AS2).

The value field is encoded as:

0								1								2								3							
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
+-----+-----+-----+-----+																															
~ Destination-IP-Origin-AS (4 octets) ~																															
+-----+-----+-----+-----+																															

Per section 10 of [RFC8955] , If a receiving BGP speaker cannot support this new Flow Specification component type, it MUST discard the NLRI value field that contains such unknown components. Since the NLRI field encoding (Section 4 of [RFC8955]) is defined in the form of a 2-tuple <length, NLRI value>, message decoding can skip over the unknown NLRI value and continue with subsequent remaining NLRI.

4. Use Case

This section describes how to use this function in a simple scenario. Considering the topology shown in Figure 1. In AS64597's R1, if the ISP AS64597 wants to redirect all packets originating from IP Prefix 61 to AS64598, first go to R3, then forward them to AS64598", the ISP AS64597 can use the traditional method or the method defining in this draft.

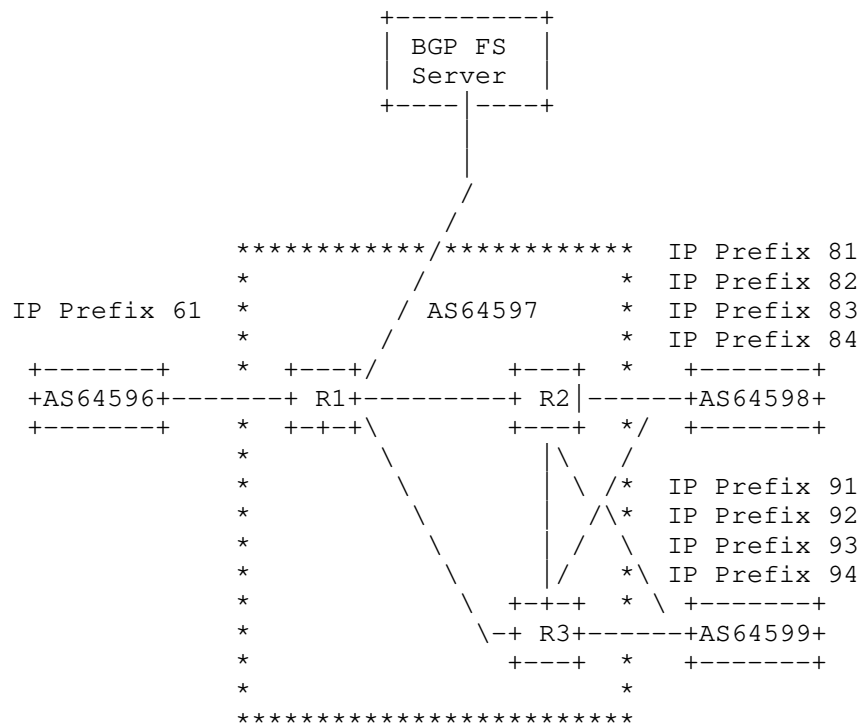


Figure 1: Redirect the traffic using Flowspec

Using the traditional method, the ISP AS64597 needs to setup multiple "Destination Prefix + Source Prefix" rules in Router R1 as following:

Destination Prefix	Source Prefix	Redirect to IP Nexthop
IP Prefix 81	IP Prefix 61	R3
IP Prefix 82	IP Prefix 61	R3
IP Prefix 83	IP Prefix 61	R3
IP Prefix 84	IP Prefix 61	R3
More...		

Figure 2: Using the traditional method to redirect the traffic

Using the method defining in this draft, the ISP AS64597 needs to setup only one "Destination Origin AS + Source Prefix" rule in Router R1 as following:

Destination IP Origin AS	Source Prefix	Redirect to IP Nexthop
64598	IP Prefix 61	R3

Figure 3: Using the AS-level filtering method to redirect the traffic

Obviously, the new method defining in this draft saves a lot of entry spaces on the control plane and forwarding plane, and it would greatly simplify the operation of the control plane, and the more destination prefixes an AS has, the more obvious the benefit.

5. Security Considerations

No new security issues are introduced to the BGP protocol by this specification.

6. IANA

IANA is requested to a new entry in "Flow Spec component types registry" with the following values:

Type	RFC or Draft	Description
TBD1	This Draft	Destination-IP-Origin-AS

7. Contributors

TBD

8. Acknowledgments

The authors would like to acknowledge the review and inputs from Gang Yan, Zhenbin Li, Rainbow Wu, Jie Dong and Ziqing Cao.

9. References

- [I-D.ietf-idr-flowspec-l2vpn]
Hao, W., Eastlake, D. E., Litkowski, S., and S. Zhuang,
"BGP Dissemination of L2 Flow Specification Rules", draft-
ietf-idr-flowspec-l2vpn-18 (work in progress), October
2021.
- [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>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A
Border Gateway Protocol 4 (BGP-4)", RFC 4271,
DOI 10.17487/RFC4271, January 2006,
<<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC8955] Loibl, C., Hares, S., Raszuk, R., McPherson, D., and M.
Bacher, "Dissemination of Flow Specification Rules",
RFC 8955, DOI 10.17487/RFC8955, December 2020,
<<https://www.rfc-editor.org/info/rfc8955>>.
- [RFC8956] Loibl, C., Ed., Raszuk, R., Ed., and S. Hares, Ed.,
"Dissemination of Flow Specification Rules for IPv6",
RFC 8956, DOI 10.17487/RFC8956, December 2020,
<<https://www.rfc-editor.org/info/rfc8956>>.

Authors' Addresses

Haibo Wang
Huawei
156 Beiqing Road
Beijing 100095
P.R. China

Email: rainsword.wang@huawei.com

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing 102209
P.R. China

Email: wangaj3@chinatelecom.cn

Shunwan Zhuang
Huawei
156 Beiqing Road
Beijing 100095
P.R. China

Email: zhuangshunwan@huawei.com

IDR Working Group
Internet-Draft
Intended status: Standards Track
Expires: 20 October 2022

H. Chen
Futurewei
Z. Li
Huawei
Z. Li
China Mobile
Y. Fan
Casa Systems
M. Toy
Verizon
L. Liu
Fujitsu
18 April 2022

BGP Extensions for IDs Allocation
draft-wu-idr-bgp-segment-allocation-ext-09

Abstract

This document describes extensions to the BGP for IDs allocation. The IDs are SIDs for segment routing (SR), including SR for IPv6 (SRv6). They are distributed to their domains if needed.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 October 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Protocol Extensions	4
3.1. Node SID NLRI TLV	4
3.2. Link SID NLRI TLV	7
3.3. Prefix SID NLRI TLV	10
3.4. Capability Negotiation	11
4. IANA Considerations	11
5. Security Considerations	12
6. Acknowledgements	13
7. References	13
7.1. Normative References	13
7.2. Informative References	15
Authors' Addresses	15

1. Introduction

In a network with a central controller, the controller has the link state information of the network, including the resource such as traffic engineering and SIDs information. It is valuable for the controller to allocate and manage the resources including SIDs of the network in a centralized way, especially for the SIDs representing network resources [I-D.ietf-teas-enhanced-vpn].

When BGP as a controller allocates an ID, it is natural and beneficial to extend BGP to send it to its corresponding network elements.

PCE may be extended to send IDs to their corresponding network elements after the IDs are allocated by a controller. However, when BGP is already deployed in a network, using PCE for IDs will need to deploy an extra protocol PCE in the network. This will increase the CapEx and OpEx.

Yang may be extended to send IDs to their corresponding network elements after the IDs are allocated by a controller. However, Yang progress may be slow. Some people may not like this.

There may not be these issues when BGP is used to send IDs. In addition, BGP may be used to distribute IDs into their domains easily when needed. It is also fit for the dynamic and static allocation of IDs.

This document proposes extensions to the BGP for sending Segment Identifiers (SIDs) for segment routing (SR) including SRv6 to their corresponding network elements after SIDs are allocated by the controller. If needed, they will be distributed into their network domains.

2. Terminology

The following terminology is used in this document.

SR: Segment Routing.

SRv6: SR for IPv6

SID: Segment Identifier.

IID: Indirection Identifier.

SR-Path: Segment Routing Path.

SR-Tunnel: Segment Routing Tunnel.

RR: Route Reflector.

MPP: MPLS Path Programming.

NAI: Node or Adjacency Identifier.

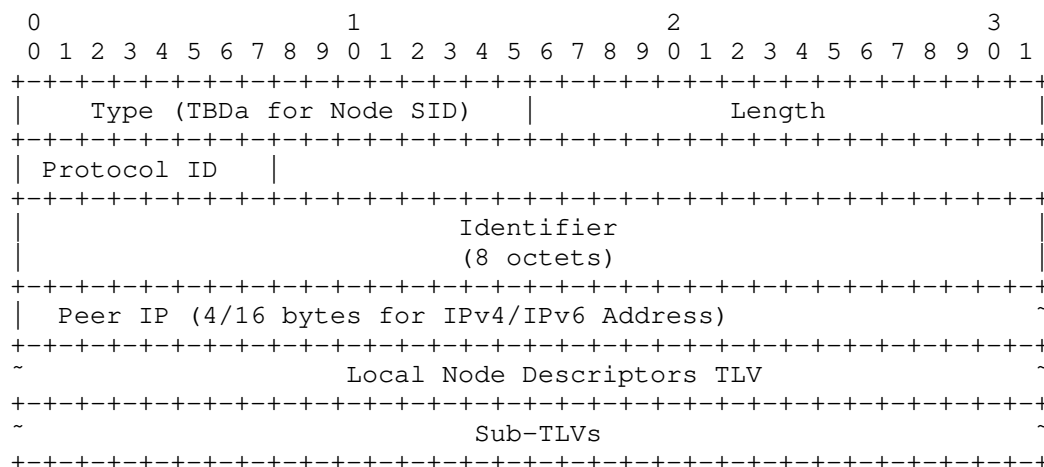
TED: Traffic Engineering Database.

3. Protocol Extensions

A new AFI and SAFI are defined: the Identifier AFI and the SID SAFI whose codepoints are to be assigned by IANA. A few new NLRI TLVs are defined for the new AFI/SAFI, which are Node, Link and Prefix SID NLRI TLVs. When a SID for a node, link or prefix is allocated by the controller, it may be sent to a network element in a UPDATE message containing a MP_REACH NLRI with the new AFI/SAFI and the SID NLRI TLV. When the SID is withdrawn by the controller, a UPDATE message containing a MP_UNREACH NLRI with the new AFI/SAFI and the SID NLRI TLV may be sent to the network element.

3.1. Node SID NLRI TLV

The Node SID NLRI TLV is used to represent the IDs such as SID associated with a node. Its format is illustrated in the Figure below, which is similar to the corresponding one defined in [RFC7752].



Where:

Type (TBDA): It is to be assigned by IANA.

Length: It is the length of the value field in bytes.

Peer IP: 4/16 octet value indicates an IPv4/IPv6 peer. When receiving a UPDATE message, a BGP speaker processes it only if the peer IP is the IP address of the BGP speaker or 0.

Protocol-ID, Identifier, and Local Node Descriptor: defined in [RFC7752], can be reused.

Sub-TLVs may be some of the followings:

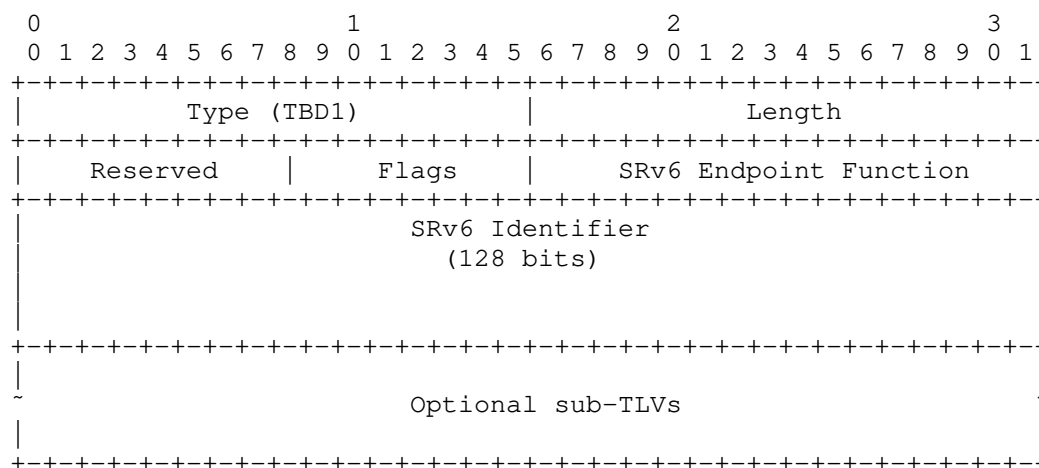
SR-Capabilities TLV (1034): It contains the Segment Routing Global Base (SRGB) range(s) allocated for the node.

SR Local Block TLV (1036): The SR Local Block (SRLB) TLV contains the range(s) of SIDs/labels allocated to the node for local SIDs.

SRv6 SID Node TLV (TBD1): A new TLV, called SRv6 Node SID TLV, contains an SRv6 SID and related information.

SRv6 Locator TLV (TBD2): A new TLV, called SRv6 Locator TLV, contains an SRv6 locator and related information.

The format of SRv6 SID Node TLV is illustrated below.



SRv6 Node SID TLV

Type: TBD1 for SRv6 Node SID TLV is to be assigned by IANA.

Length: Variable.

Flags: 1 octet. No flags are defined now.

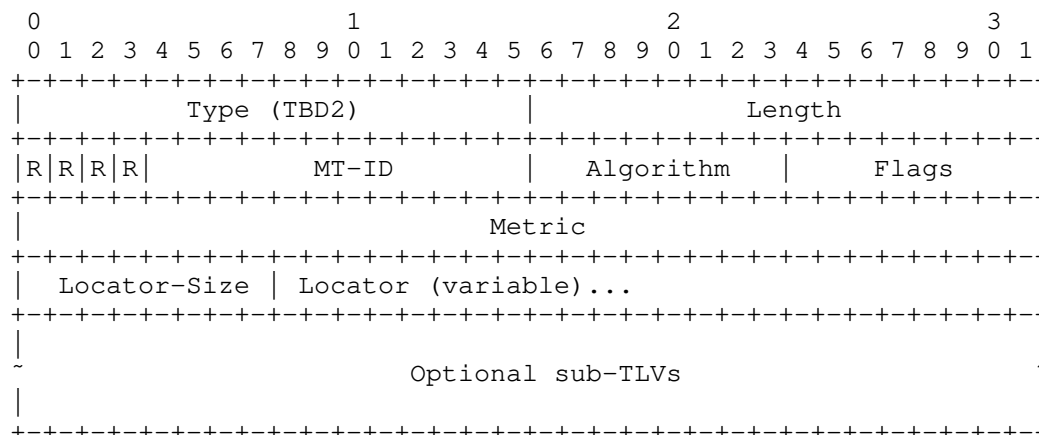
SRv6 Endpoint Function: 2 octets. The function associated with SRv6 SID.

SRv6 Identifier: 16 octets. IPv6 address representing SRv6 SID.

Reserved: MUST be set to 0 while sending and ignored on receipt.

SRv6 node SID inherits the topology and algorithm from its locator.

The format of SRv6 locator TLV is illustrated below.



SRv6 Locator TLV

Type: TBD2 for SRv6 Locator TLV is to be assigned by IANA.

Length: Variable.

MT-ID: Multitopology Identifier as defined in [RFC5120].

Algorithm: 1 octet. Associated algorithm.

Flags: 1 octet. As described in
[I-D.ietf-lsr-isis-srv6-extensions].

Metric: 4 octets. As described in [RFC5305].

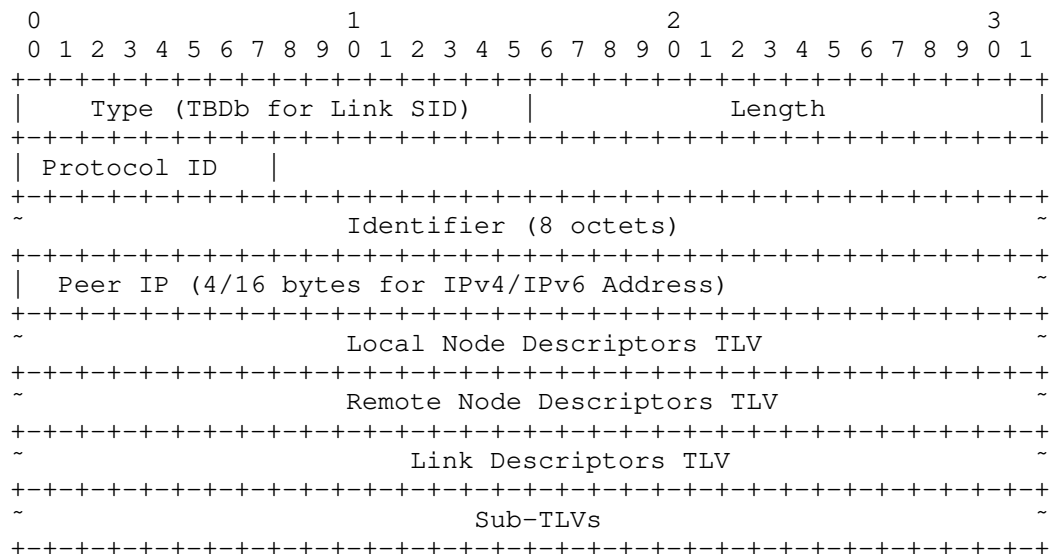
Locator-Size: 1 octet. Number of bits in the Locator field (1 to 128).

Locator: 1 to 16 octets. SRv6 Locator encoded in the minimum number of octets for the given Locator-Size.

Reserved: MUST be set to 0 while sending and ignored on receipt.

3.2. Link SID NLRI TLV

The Link SID NLRI TLV is used to represent the IDs such as SID associated with a link. Its format is illustrated in the Figure below, which is similar to the corresponding one defined in [RFC7752].



Where:

Type (TBD): It is to be assigned by IANA.

Length: It is the length of the value field in bytes.

Peer IP: 4/16 octet value indicates an IPv4/IPv6 peer.

Protocol-ID, Identifier, Local Node Descriptors, Remote Node Descriptors and Link Descriptors: defined in [RFC7752], can be reused.

The Sub-TLVs may be some of the followings:

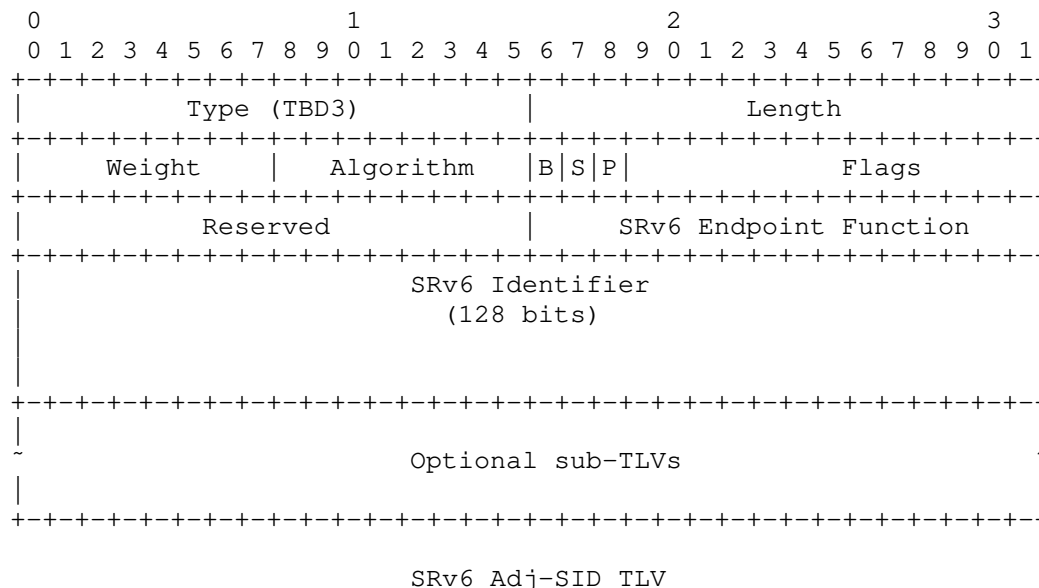
Adj-SID TLV (1099): It contains the Segment Identifier (SID) allocated for the link/adjacency.

LAN Adj-SID TLV (1100): It contains the Segment Identifier (SID) allocated for the adjacency/link to a non-DR router on a broadcast, NBMA, or hybrid link.

SRv6 Adj-SID TLV (TBD3): A new TLV, called SRv6 Adj-SID TLV, contains an SRv6 Adj-SID and related information.

SRv6 LAN Adj-SID TLV (TBD4): A new TLV, called SRv6 LAN Adj-SID TLV, contains an SRv6 LAN Adj-SID and related information.

The format of an SRv6 Adj-SID TLV is illustrated below.



Type: TBD3 for SRv6 Adj-SID TLV is to be assigned by IANA.

Length: Variable.

Weight: 1 octet. The value represents the weight of the SID for the purpose of load balancing.

Algorithm: 1 octet. Associated algorithm.

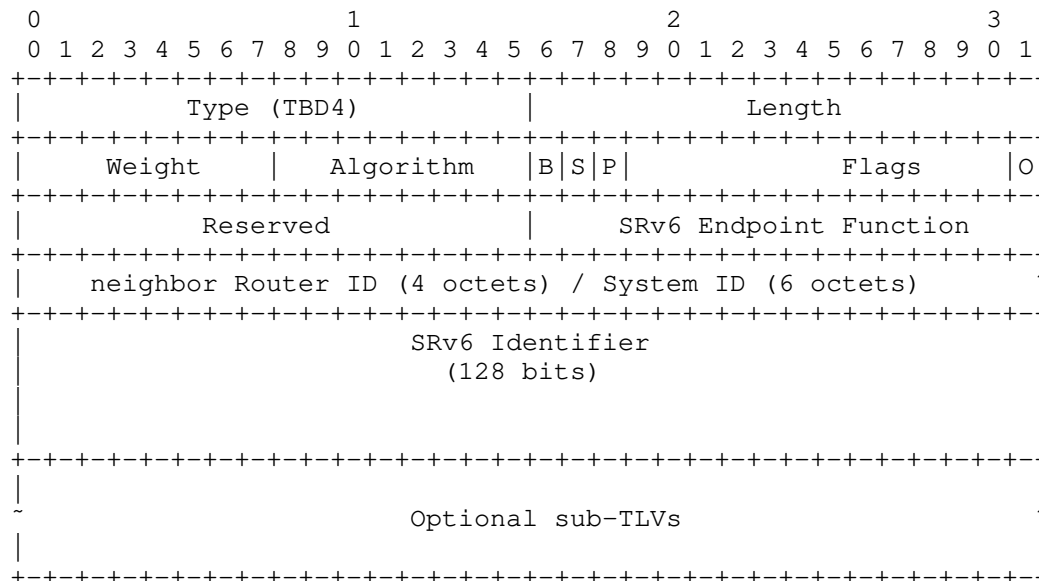
Flags: 2 octets. Three flags are defined in [I-D.ietf-lsr-isis-srv6-extensions].

SRv6 Endpoint Function: 2 octets. The function associated with SRv6 SID.

SRv6 Identifier: 16 octets. IPv6 address representing SRv6 SID.

Reserved: MUST be set to 0 while sending and ignored on receipt.

The format of an SRv6 LAN Adj-SID TLV is illustrated below.



SRv6 LAN Adj-SID TLV

Type: TBD4 for SRv6 LAN Adj-SID TLV is to be assigned by IANA.

Length: Variable.

Weight: 1 octet. The value represents the weight of the SID for the purpose of load balancing.

Algorithm: 1 octet. Associated algorithm.

Flags: 2 octets. Three flags B, S and P are defined in [I-D.ietf-lsr-isis-srv6-extensions]. Flag O set to 1 indicating OSPF neighbor Router ID of 4 octets, set to 0 indicating IS-IS neighbor System ID of 6 octets.

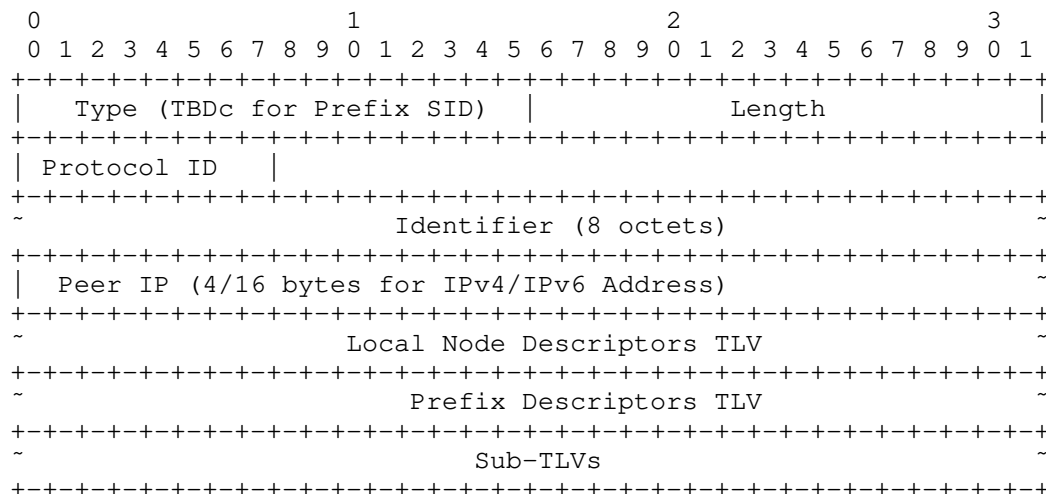
SRv6 Endpoint Function: 2 octets. The function associated with SRv6 SID.

SRv6 Identifier: 16 octets. IPv6 address representing SRv6 SID.

Reserved: MUST be set to 0 while sending and ignored on receipt.

3.3. Prefix SID NLRI TLV

The Prefix SID NLRI TLV is used to represent the IDs such as SID associated with a prefix. Its format is illustrated in the Figure below, which is similar to the corresponding one defined in [RFC7752].



Where:

Type (TBDC): It is to be assigned by IANA.

Length: It is the length of the value field in bytes.

Peer IP: 4/16 octet value indicates an IPv4/IPv6 peer.

Protocol-ID, Identifier, Local Node Descriptors and Prefix Descriptors: defined in [RFC7752], can be reused.

Sub-TLVs may be some of the followings:

Prefix-SID TLV (1158): It contains the Segment Identifier (SID) allocated for the prefix.

Prefix Range TLV (1159): It contains a range of prefixes and the Segment Identifier (SID)s allocated for the prefixes.

3.4. Capability Negotiation

It is necessary to negotiate the capability to support BGP Extensions for sending and receiving Segment Identifiers (SIDs). The BGP SID Capability is a new BGP capability [RFC5492]. The Capability Code for this capability is to be specified by the IANA. The Capability Length field of this capability is variable. The Capability Value field consists of one or more of the following tuples:

Address Family Identifier (2 octets)
Subsequent Address Family Identifier (1 octet)
Send/Receive (1 octet)

BGP SID Capability

The meaning and use of the fields are as follows:

Address Family Identifier (AFI): This field is the same as the one used in [RFC4760].

Subsequent Address Family Identifier (SAFI): This field is the same as the one used in [RFC4760].

Send/Receive: This field indicates whether the sender is (a) willing to receive SID from its peer (value 1), (b) would like to send SID to its peer (value 2), or (c) both (value 3) for the <AFI, SAFI>.

4. IANA Considerations

This document requests assigning a new AFI in the registry "Address Family Numbers" as follows:

Code Point	Description	Reference
TBDx	Identifier AFI	This document

This document requests assigning a new SAFI in the registry "Subsequent Address Family Identifiers (SAFI) Parameters" as follows:

Code Point	Description	Reference
TBDy	SID SAFI	This document

This document defines a new registry called "SID NLRI TLVs". The allocation policy of this registry is "First Come First Served (FCFS)" according to [RFC8126].

Following TLV code points are defined:

Code Point	Description	Reference
1 (TBDA)	Node SID NLRI	This document
2 (TBDB)	Link SID NLRI	This document
3 (TBDC)	Prefix SID NLRI	This document

This document requests assigning a code-point from the registry "BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs" as follows:

TLV Code Point	Description	Reference
TBD1	SRv6 Node SID	This document
TBD2	SRv6 Allocator	This document
TBD3	SRv6 Adj-SID	This document
TBD4	SRv6 LAN Adj-SID	This document

5. Security Considerations

Protocol extensions defined in this document do not affect the BGP security other than those as discussed in the Security Considerations section of [RFC7752].

6. Acknowledgements

The authors would like to thank Eric Wu, Robert Raszuk, Zhengquiang Li, and Ketan Talaulikar for their valuable suggestions and comments on this draft.

7. References

7.1. Normative References

- [I-D.ietf-idr-flowspec-path-redirect]
Velde, G. V. D., Patel, K., and Z. Li, "Flowspec Indirection-id Redirect", Work in Progress, Internet-Draft, draft-ietf-idr-flowspec-path-redirect-11, 26 May 2020, <<https://www.ietf.org/archive/id/draft-ietf-idr-flowspec-path-redirect-11.txt>>.
- [I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", Work in Progress, Internet-Draft, draft-ietf-isis-segment-routing-extensions-25, 19 May 2019, <<https://www.ietf.org/archive/id/draft-ietf-isis-segment-routing-extensions-25.txt>>.
- [I-D.ietf-lsr-isis-srv6-extensions]
Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Segment Routing over IPv6 Dataplane", Work in Progress, Internet-Draft, draft-ietf-lsr-isis-srv6-extensions-18, 20 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-lsr-isis-srv6-extensions-18.txt>>.
- [I-D.ietf-rtgwg-bgp-routing-large-dc]
Lapukhov, P., Premji, A., and J. Mitchell, "Use of BGP for Routing in Large-Scale Data Centers", Work in Progress, Internet-Draft, draft-ietf-rtgwg-bgp-routing-large-dc-11, 4 June 2016, <<https://www.ietf.org/archive/id/draft-ietf-rtgwg-bgp-routing-large-dc-11.txt>>.
- [I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-15, 25 January 2018, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-15.txt>>.

- [I-D.ietf-spring-segment-routing-ldp-interop]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., and S. Litkowski, "Segment Routing MPLS Interworking with LDP", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-ldp-interop-15, 2 September 2018, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-ldp-interop-15.txt>>.
- [I-D.li-ospf-ospfv3-srv6-extensions]
Li, Z., Hu, Z., Cheng, D., Talaulikar, K., and P. Psenak, "OSPFv3 Extensions for SRv6", Work in Progress, Internet-Draft, draft-li-ospf-ospfv3-srv6-extensions-07, 4 November 2019, <<https://www.ietf.org/archive/id/draft-li-ospf-ospfv3-srv6-extensions-07.txt>>.
- [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>>.
- [RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <<https://www.rfc-editor.org/info/rfc4760>>.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <<https://www.rfc-editor.org/info/rfc5120>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC5492] Scudder, J. and R. Chandra, "Capabilities Advertisement with BGP-4", RFC 5492, DOI 10.17487/RFC5492, February 2009, <<https://www.rfc-editor.org/info/rfc5492>>.
- [RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J., and D. McPherson, "Dissemination of Flow Specification Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009, <<https://www.rfc-editor.org/info/rfc5575>>.

- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<https://www.rfc-editor.org/info/rfc7752>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.

7.2. Informative References

- [I-D.gredler-idr-bgp-ls-segment-routing-extension]
Gredler, H., Ray, S., Previdi, S., Filsfils, C., Chen, M., and J. Tantsura, "BGP Link-State extensions for Segment Routing", Work in Progress, Internet-Draft, draft-gredler-idr-bgp-ls-segment-routing-extension-02, 16 October 2014, <<https://www.ietf.org/archive/id/draft-gredler-idr-bgp-ls-segment-routing-extension-02.txt>>.
- [I-D.ietf-idr-bgppls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "Border Gateway Protocol - Link State (BGP-LS) Extensions for Segment Routing BGP Egress Peer Engineering", Work in Progress, Internet-Draft, draft-ietf-idr-bgppls-segment-routing-epe-19, 16 May 2019, <<https://www.ietf.org/archive/id/draft-ietf-idr-bgppls-segment-routing-epe-19.txt>>.
- [I-D.ietf-teas-enhanced-vpn]
Dong, J., Bryant, S., Li, Z., Miyasaka, T., and Y. Lee, "A Framework for Enhanced Virtual Private Network (VPN+) Services", Work in Progress, Internet-Draft, draft-ietf-teas-enhanced-vpn-10, 6 March 2022, <<https://www.ietf.org/archive/id/draft-ietf-teas-enhanced-vpn-10.txt>>.

Authors' Addresses

Huaimo Chen
Futurewei
Boston, MA,
United States of America
Email: Huaimo.chen@futurewei.com

Zhenbin Li
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing
100095
China
Email: lizhenbin@huawei.com

Zhenqiang Li
China Mobile
No. 29 Finance Street, Xicheng District
Beijing
100029
P.R. China
Email: li_zhenqiang@hotmail.com

Yanhe Fan
Casa Systems
United States of America
Email: yfan@casa-systems.com

Mehmet Toy
Verizon
United States of America
Email: mehmet.toy@verizon.com

Lei Liu
Fujitsu
United States of America
Email: liulei.kddi@gmail.com