

IDR Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2020

H. Chen
Futurewei
Z. Li
S. Zhuang
Huawei
July 7, 2019

BGP Extensions for SRv6 SIDs Allocation
draft-chen-idr-bgp-srv6-sid-allocation-01

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 January 8, 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. Terminology | 3 |
| 3. Protocol Extensions | 3 |
| 3.1. Node SID NLRI TLV | 4 |
| 3.2. Link SID NLRI TLV | 6 |
| 3.3. Prefix SID NLRI TLV | 10 |
| 4. IANA Considerations | 11 |
| 5. Security Considerations | 11 |
| 6. Acknowledgements | 12 |
| 7. References | 12 |
| 7.1. Normative References | 12 |
| 7.2. Informative References | 13 |
| Authors' Addresses | 14 |

1. Introduction

In a network with a central controller, the controller has the link state information of the network, including traffic engineering information. In addition, the controller allocates and manages the resources of the network in general. It is natural and beneficial for the controller to allocate and manage IDs as a kind of network resources.

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.

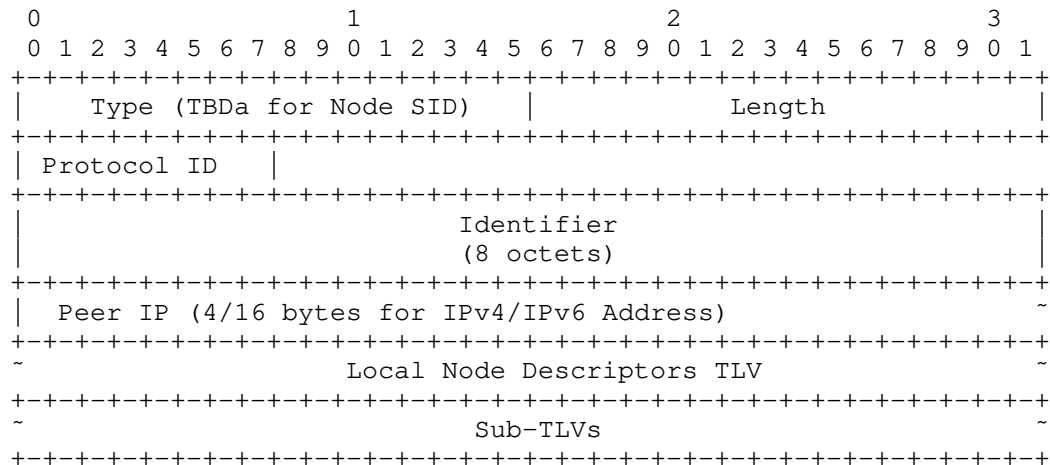
TED: Traffic Engineering Database.

3. Protocol Extensions

A new SAFI is defined: the SID SAFI whose codepoint TBD is to be assigned by IANA. A few new NLRI TLVs are defined for the new SAFI, which are Node, Link and Prefix SID NLRI TLV. 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 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 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 for allocating 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:

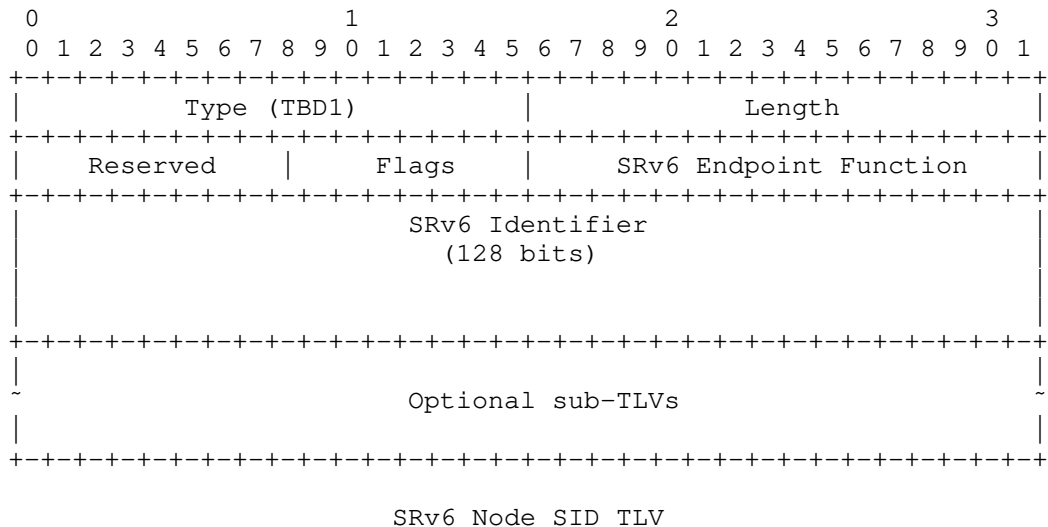
SR-Capabilities TLV (1034): It contains the Segment Routing Global Base (SRGB) range(s) allocated for the node.

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.



Type: TBD1 for SRv6 Node SID TLV is to be assigned by IANA.

Length: Variable.

Flags: 1 octet. No flags are defined now.

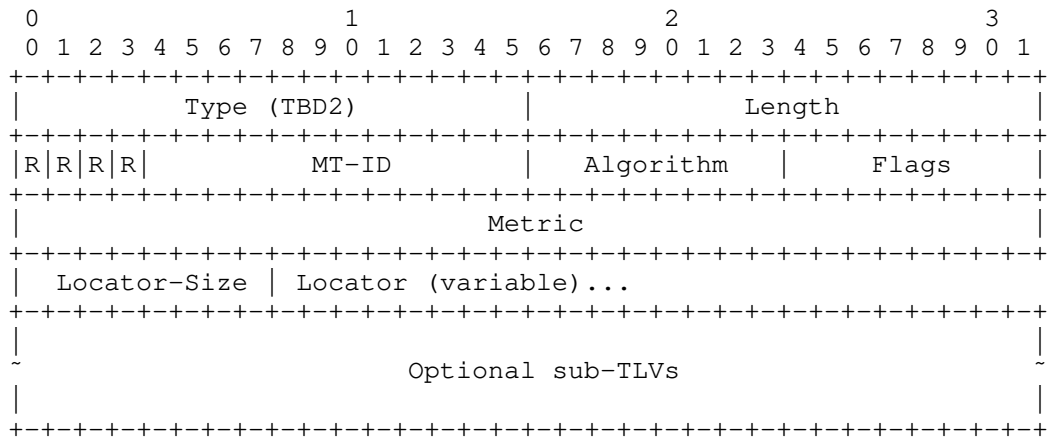
SRv6 Endpoint Function: 2 octets. The function associated with SRv6 SID.

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.bashandy-isis-srv6-extensions].

Metric: 4 octets. As described in [RFC5305].

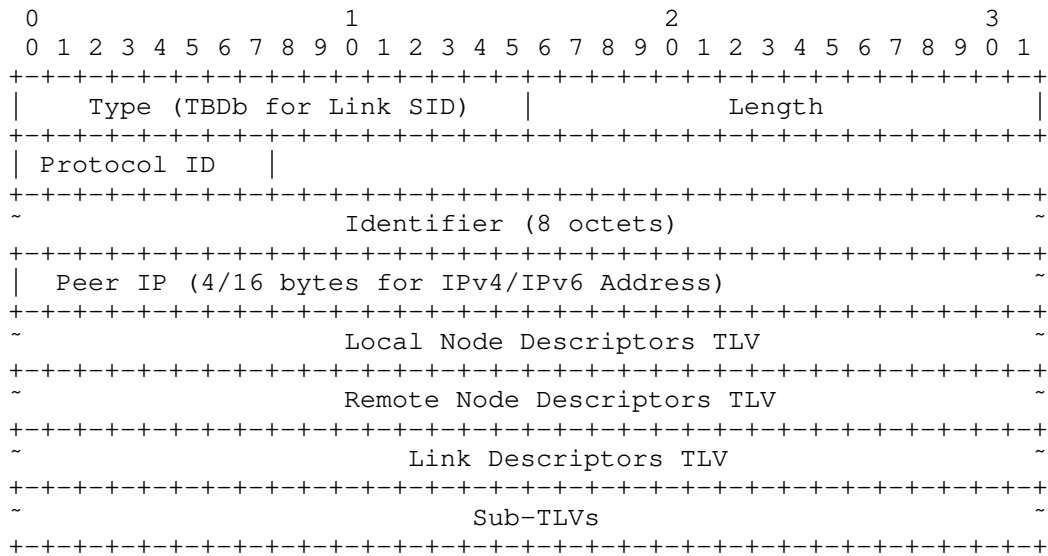
Locator-Size: 1 octet. Number of bits in the Locator field (1 to 128).

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 for allocating 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_a): 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:

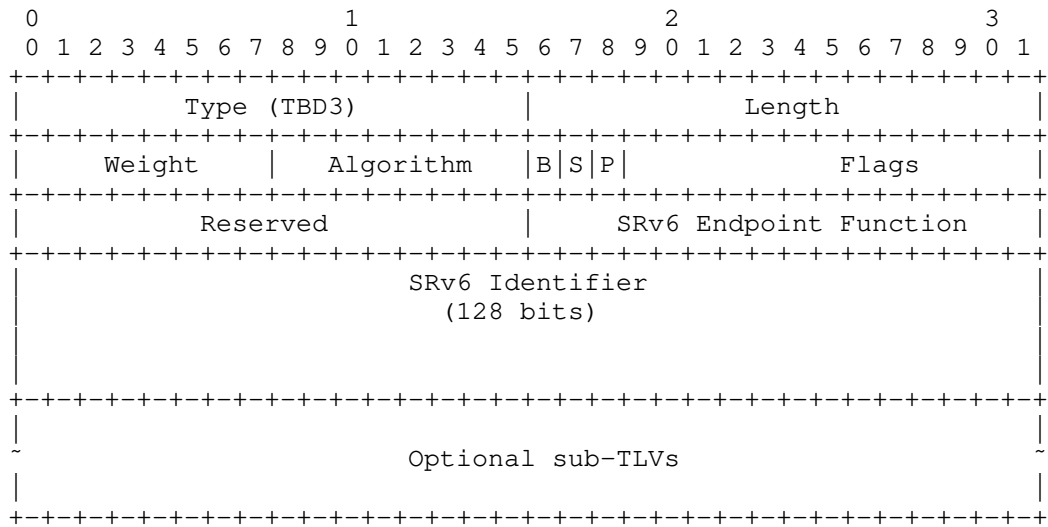
Adj-SID TLV (1099): It contains the Segment Identifier (SID) allocated for the link/adjacency.

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.



SRv6 Adj-SID TLV

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.

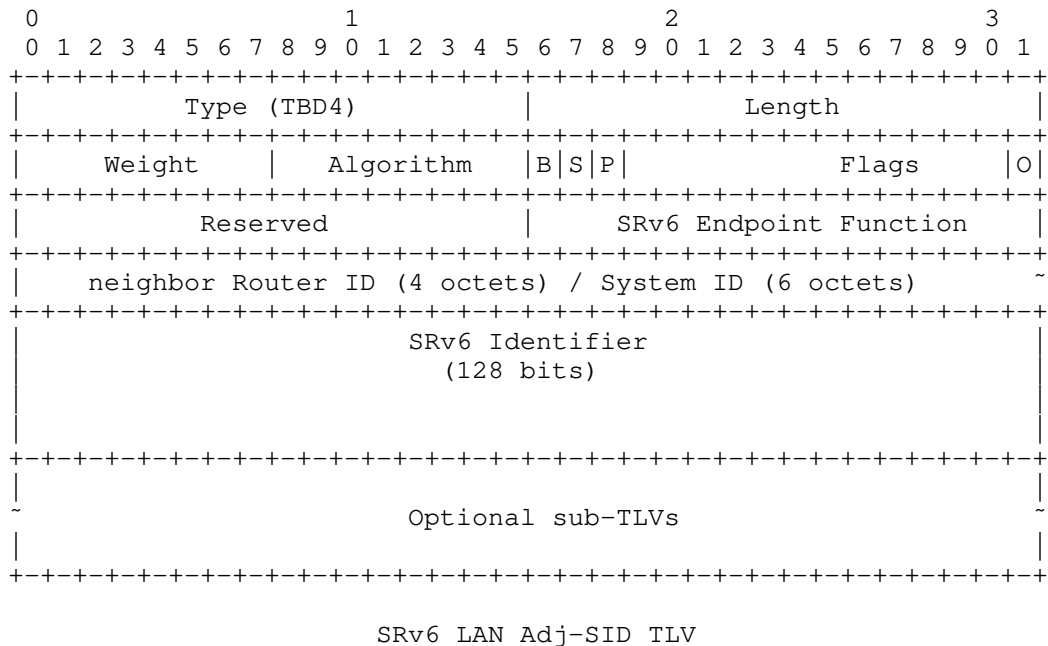
Flags: 2 octets. Three flags are defined in [I-D.bashandy-isis-srv6-extensions].

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.



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.bashandy-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.

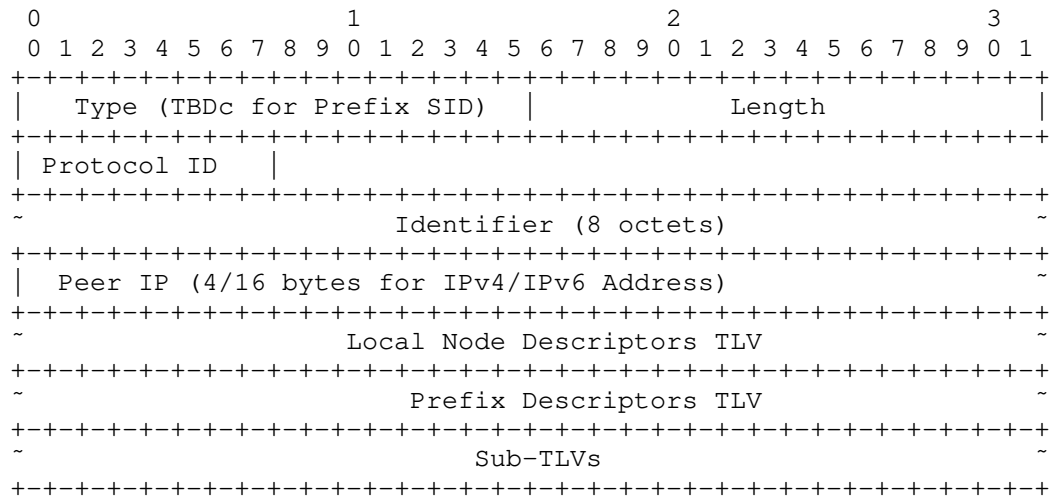
SRv6 Endpoint Function: 2 octets. The function associated with SRv6 SID.

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 for allocating 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.

Prefix IDs Allocation field may contain 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.

4. IANA Considerations

This document requests assigning a new SAFI in the registry "Subsequent Address Family Identifiers (SAFI) Parameters" as follows:

| Code Point | Description | Reference |
|------------|-------------|---------------|
| TBD | 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 Razuk, Zhengquiang Li, and Ketan Talaulikar for their valuable suggestions and comments on this draft.

7. References

7.1. Normative References

- [I-D.bashandy-isis-srv6-extensions]
Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Routing over IPv6 Dataplane", draft-bashandy-isis-srv6-extensions-05 (work in progress), March 2019.
- [I-D.ietf-idr-flowspec-path-redirect]
Velde, G., Patel, K., and Z. Li, "Flowspec Indirection-id Redirect", draft-ietf-idr-flowspec-path-redirect-08 (work in progress), June 2019.
- [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", draft-ietf-isis-segment-routing-extensions-25 (work in progress), May 2019.
- [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", draft-ietf-rtgwg-bgp-routing-large-dc-11 (work in progress), June 2016.
- [I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-15 (work in progress), January 2018.
- [I-D.ietf-spring-segment-routing-ldp-interop]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., and S. Litkowski, "Segment Routing interworking with LDP", draft-ietf-spring-segment-routing-ldp-interop-15 (work in progress), September 2018.
- [I-D.li-ospf-ospfv3-srv6-extensions]
Li, Z., Hu, Z., Cheng, D., Talaulikar, K., and P. Psenak, "OSPFv3 Extensions for SRv6", draft-li-ospf-ospfv3-srv6-extensions-03 (work in progress), March 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>>.
- [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>>.
- [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", draft-gredler-idr-bgp-ls-segment-routing-extension-02 (work in progress), October 2014.
- [I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-19 (work in progress), May 2019.

Authors' Addresses

Huaimo Chen
Futurewei
Boston, MA
USA

Email: Huaimo.chen@futurewei.com

Zhenbin Li
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Shunwan Zhuang
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: zhuangshunwan@huawei.com

Inter-Domain Routing
Internet-Draft
Intended status: Standards Track
Expires: September 25, 2019

G. Dawra, Ed.
LinkedIn
C. Filsfils
K. Talaulikar, Ed.
Cisco Systems
M. Chen
Huawei
D. Bernier
Bell Canada
J. Uttaro
AT&T
B. Decraene
Orange
H. Elmalky
Ericsson
March 24, 2019

BGP Link State Extensions for SRv6
draft-dawra-idr-bgpls-srv6-ext-06

Abstract

Segment Routing IPv6 (SRv6) allows for a flexible definition of end-to-end paths within various topologies by encoding paths as sequences of topological or functional sub-paths, called "segments". These segments are advertised by the various protocols such as BGP, ISIS and OSPFv3.

BGP Link-state (BGP-LS) address-family solution for SRv6 is similar to BGP-LS for SR for MPLS dataplane. This draft defines extensions to the BGP-LS to advertise SRv6 Segments along with their functions and other attributes via BGP.

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 September 25, 2019.

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 | 3 |
| 2. BGP-LS Extensions for SRv6 | 4 |
| 3. SRv6 Node Attributes | 5 |
| 3.1. SRv6 Capabilities TLV | 5 |
| 3.2. SRv6 Node MSD Types | 6 |
| 4. SRv6 Link Attributes | 7 |
| 4.1. SRv6 End.X SID TLV | 7 |
| 4.2. SRv6 LAN End.X SID TLV | 9 |
| 4.3. SRv6 Link MSD Types | 11 |
| 5. SRv6 Prefix Attributes | 12 |
| 5.1. SRv6 Locator TLV | 12 |
| 6. SRv6 SID NLRI | 14 |
| 6.1. SRv6 SID Information TLV | 15 |
| 7. SRv6 SID Attributes | 16 |
| 7.1. SRv6 Endpoint Function TLV | 16 |
| 7.2. SRv6 BGP Peer Node SID TLV | 17 |
| 8. IANA Considerations | 19 |
| 8.1. BGP-LS NLRI-Types | 19 |
| 8.2. BGP-LS TLVs | 19 |

| | |
|---|----|
| 9. Manageability Considerations | 20 |
| 10. Operational Considerations | 20 |
| 10.1. Operations | 20 |
| 11. Security Considerations | 20 |
| 12. Contributors | 20 |
| 13. Acknowledgements | 21 |
| 14. References | 21 |
| 14.1. Normative References | 21 |
| 14.2. Informative References | 23 |
| Authors' Addresses | 23 |

1. Introduction

SRv6 refers to Segment Routing instantiated on the IPv6 dataplane [RFC8402]. Segment Identifier (SID) is often used as a shorter reference for "SRv6 Segment".

The network programming paradigm [I-D.filsfils-spring-srv6-network-programming] is central to SRv6. It describes how different functions can be bound to their SIDs and how a network program can be expressed as a combination of SIDs.

An SRv6-capable node N maintains a "My SID Table" (refer [I-D.filsfils-spring-srv6-network-programming]). This table contains all the SRv6 segments explicitly instantiated at node N.

The IS-IS [I-D.bashandy-isis-srv6-extensions] and OSPFv3 [I-D.li-ospf-ospfv3-srv6-extensions] link-state routing protocols have been extended to advertise some of these SRv6 SIDs and SRv6-related information. BGP ([I-D.dawra-idr-srv6-vpn]) has been extended to advertise some of these SRv6 SIDs for VPN services. Certain other SRv6 SIDs may be instantiated on a node via other mechanisms for topological or service functionalities.

The advertisement of SR related information along with the topology for the MPLS dataplane instantiation is specified in [I-D.ietf-idr-bgp-ls-segment-routing-ext] and for the BGP Egress Peer Engineering (EPE) is specified in [I-D.ietf-idr-bgpls-segment-routing-epe]. On the similar lines, introducing the SRv6 related information in BGP-LS allows it's consumer applications that require topological visibility to also receive the "My SID Table" from nodes across a domain or even across Autonomous Systems (AS), as required. This allows applications to leverage the SRv6 capabilities for network programming.

The identifying key of each Link-State object, namely a node, link, or prefix, is encoded in the NLRI and the properties of the object are encoded in the BGP-LS Attribute [RFC7752].

This document describes extensions to BGP-LS to advertise the SRv6 "My SID Table" and other SRv6 information from all the SRv6 capable nodes in the domain when sourced from link-state routing protocols and directly from individual SRv6 capable nodes when sourced from BGP.

2. BGP-LS Extensions for SRv6

BGP-LS[RFC7752] defines the BGP Node, Link and Prefix attributes. All non-VPN link, node, and prefix information SHALL be encoded using AFI 16388 / SAFI 71. VPN link, node, and prefix information SHALL be encoded using AFI 16388 / SAFI 72.

The SRv6 information pertaining to a node is advertised via the BGP-LS Node NLRI and using the BGP-LS Attribute TLVs as follows:

- o SRv6 Capabilities of the node is advertised via a new SRv6 Capabilities TLV
- o New MSD types introduced for SRv6 are advertised as new sub-TLVs of the Node MSD TLV specified in [I-D.ietf-idr-bgp-ls-segment-routing-msd].
- o Algorithm support for SRv6 is advertised via the existing SR Algorithm TLV specified in [I-D.ietf-idr-bgp-ls-segment-routing-ext].

The SRv6 information pertaining to a link is advertised via the BGP-LS Link NLRI and using the BGP-LS Attribute TLVs as follows:

- o SRv6 End.X SID of the link state routing adjacency or the BGP EPE Peer Adjacency is advertised via a new SRv6 End.X SID TLV
- o SRv6 LAN End.X SID of the link state routing adjacency to a non-DR/DIS router is advertised via a new SRv6 LAN End.X SID TLV
- o New MSD types introduced for SRv6 are advertised as new sub-TLVs of the Link MSD TLV specified in [I-D.ietf-idr-bgp-ls-segment-routing-msd].

The SRv6 Locator information of a node is advertised via the BGP-LS Prefix NLRI using the new SRv6 Locator TLV in the BGP-LS Attribute.

The SRv6 SIDs associated with the node from its "My SID Table" are advertised as a newly introduce BGP-LS SRv6 SID NLRI. This enables the BGP-LS encoding to scale to cover a potentially large set of SRv6 SIDs instantiated on a node with the granularity of individual SIDs and without affecting the size and scalability of the BGP-LS updates.

New BGP-LS Attribute TLVs are introduced for the SRv6 SID NLRI as follows:

- o The endpoint function of the SRv6 SID is advertised via a new SRv6 Endpoint Function TLV
- o The BGP EPE Peer Node and Peer Set SID context is advertised via a new SRv6 BGP EPE Peer Node SID TLV

When the BGP-LS router is advertising topology information that it sources from the underlying link-state routing protocol, then it maps the corresponding SRv6 information from the SRv6 extensions for IS-IS [I-D.bashandy-isis-srv6-extensions] and OSPFv3 [I-D.li-ospf-ospfv3-srv6-extensions] protocols to their BGP-LS TLVs/sub-TLVs for all SRv6 capable nodes in that routing protocol domain. When the BGP-LS router is advertising topology information from the BGP routing protocol [I-D.ietf-idr-bgpls-segment-routing-epe], then it advertises the SRv6 information from the local node alone (e.g. BGP EPE topology information or in the case of a data center network running BGP as the only routing protocol).

Subsequent sections of this document specify the encoding of the newly defined extensions.

3. SRv6 Node Attributes

SRv6 attributes of a node are advertised using the new BGP-LS Attribute TLVs defined in this section and associated with the BGP-LS Node NLRI.

3.1. SRv6 Capabilities TLV

This BGP-LS Attribute TLV is used to announce the SRv6 capabilities of the node along with the BGP-LS Node NLRI and indicates the SRv6 support by the node. A single instance of this TLV MUST be included in the BGP-LS attribute for each SRv6 capable node. This TLV maps to the SRv6 Capabilities sub-TLV and the SRv6 Capabilities TLV of the IS-IS and OSPFv3 protocol SRv6 extensions respectively.

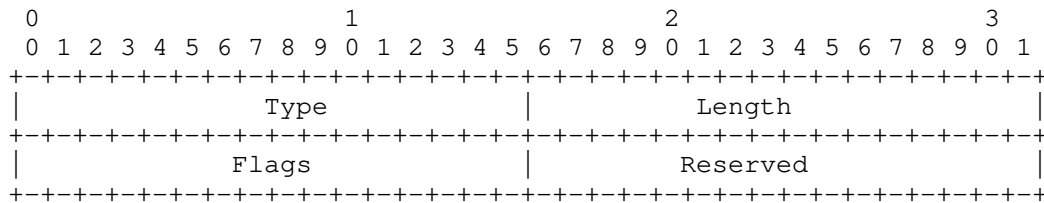


Figure 1: SRv6 Capabilities TLV Format

Where:

- o Type: 2 octet field with value TBD, see Section 8.
- o Length : 2 octet field with value set to 4.
- o Flags: 2 octet field. The following flags are defined:

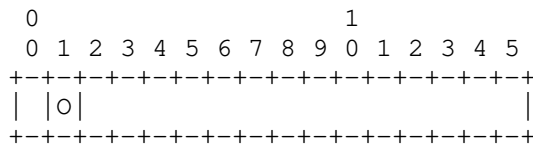


Figure 2: SRv6 Capability TLV Flags Format

- * O-flag: If set, then router is capable of supporting SRH O-bit Flags, as specified in [I-D.ali-spring-srv6-oam].
- o Reserved: 2 octet that SHOULD be set to 0 and MUST be ignored on receipt.

3.2. SRv6 Node MSD Types

The Node MSD TLV [I-D.ietf-idr-bgp-ls-segment-routing-msd] of the BGP-LS Attribute of the Node NLRI is also used to advertise the limits and the supported Segment Routing Header (SRH) [I-D.ietf-6man-segment-routing-header] operations supported by the SRv6 capable node. The SRv6 MSD Types specified in [I-D.bashandy-isis-srv6-extensions] are also used with the BGP-LS Node MSD TLV as these codepoints are shared between IS-IS, OSPF and BGP-LS protocols. The description and semantics of these new MSD types for BGP-LS are identical as specified [I-D.bashandy-isis-srv6-extensions] and summarized in the table below:

| MSD Type | Description |
|----------|-----------------------|
| TBD | Maximum Segments Left |
| TBD | Maximum End Pop |
| TBD | Maximum T.Insert |
| TBD | Maximum T.Encaps |
| TBD | Maximum End D |

Figure 3: SRv6 Node MSD Types

Each MSD type is encoded as a one octet type followed by a one octet value.

4. SRv6 Link Attributes

SRv6 attributes and SIDs associated with a link or adjacency are advertised using the new BGP-LS Attribute TLVs defined in this section and associated with the BGP-LS Link NLRI.

4.1. SRv6 End.X SID TLV

The SRv6 End.X SID TLV is used to advertise the SRv6 End.X SIDs that correspond to a point-to-point or point-to-multipoint link or adjacency of the local node for IS-IS and OSPFv3 protocols. This TLV can also be used to advertise the End.X function SRv6 SID corresponding to the underlying layer-2 member links for a layer-3 bundle interface using L2 Bundle Member Attribute TLV as specified in [I-D.ietf-idr-bgp-ls-segment-routing-ext] .

For the nodes running BGP routing protocol, this TLV is used to advertise the BGP EPE Peer Adjacency SID for SRv6 on the same lines as specified for SR/MPLS in [I-D.ietf-idr-bgpls-segment-routing-epe]. The SRv6 End.X SID for the BGP Peer Adjacency indicates the cross-connect to a specific layer-3 link to the specific BGP session peer (neighbor).

The TLV has the following format:

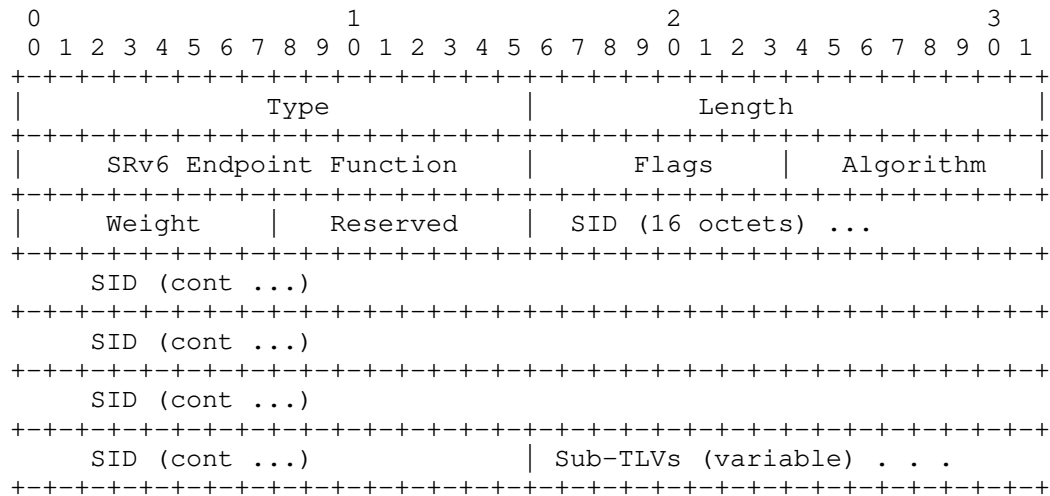


Figure 4: SRv6 End.X TLV Format

Where:

Type: 2 octet field with value TBD, see Section 8.

Length: 2 octet field with the total length of the value portion of the TLV.

Function Code: 2 octet field. The Endpoint Function code point for this SRv6 SID as defined in [I-D.filsfils-spring-srv6-network-programming].

Flags: 1 octet of flags with the following definition:

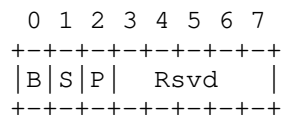


Figure 5: SRv6 End.X SID TLV Flags Format

- * B-Flag: Backup Flag. If set, the SID is eligible for protection (e.g. using IPFRR) as described in [RFC8355].
- * S-Flag: Set Flag. When set, the S-Flag indicates that the SID refers to a set of adjacencies (and therefore MAY be assigned to other adjacencies as well).

- * P-Flag: Persistent Flag: When set, the P-Flag indicates that the SID is persistently allocated, i.e., the value remains consistent across router restart and/or interface flap.
- * Rsvd bits: Reserved for future use and MUST be zero when originated and ignored when received.

Algorithm: 1 octet field. Algorithm associated with the SID. Algorithm values are defined in the IGP Algorithm Type registry.

Weight: 1 octet field. The value represents the weight of the SID for the purpose of load balancing. The use of the weight is defined in [RFC8402].

Reserved: 1 octet field that SHOULD be set to 0 and MUST be ignored on receipt.

SID: 16 octet field. This field encodes the advertised SRv6 SID as 128 bit value.

Sub-TLVs : currently none defined. Used to advertise sub-TLVs that provide additional attributes for the given SRv6 End.X SID.

4.2. SRv6 LAN End.X SID TLV

For a LAN interface, normally a node only announces its adjacency to the IS-IS pseudo-node (or the equivalent OSPF Designated Router). The SRv6 LAN End.X SID TLV allows a node to announce SRv6 SID corresponding to functions like END.X for its adjacencies to all other (i.e. non-DIS or non-DR) nodes attached to the LAN in a single instance of the BGP-LS Link NLRI. Without this TLV, the corresponding BGP-LS link NLRI would need to be originated for each additional adjacency in order to advertise the SRv6 End.X SID TLVs for these neighbor adjacencies.

The IS-IS and OSPFv3 SRv6 LAN End.X SID TLVs have the following format:

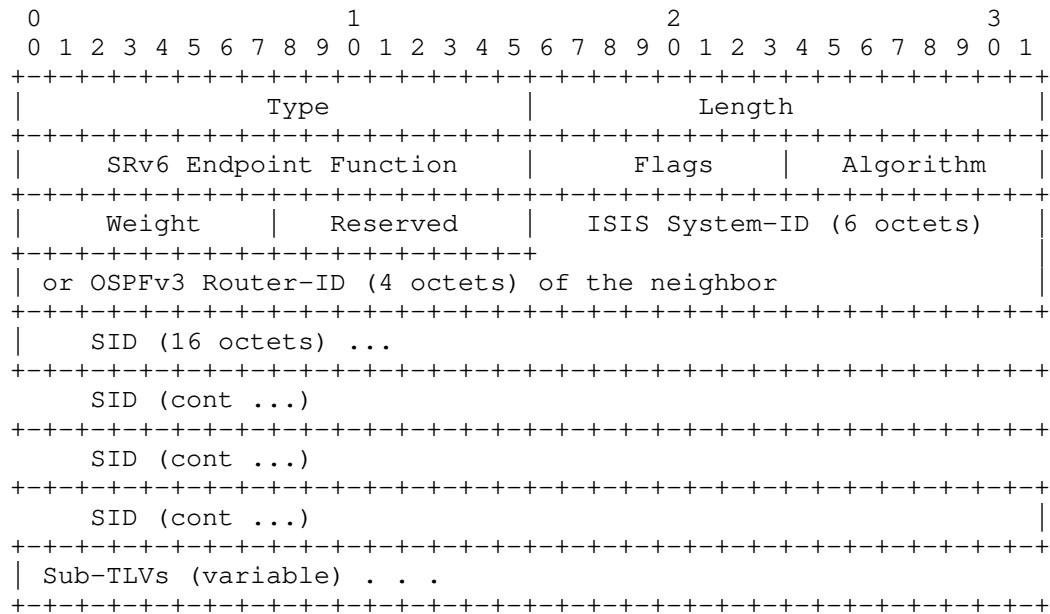


Figure 6: SRv6 LAN End.X SID TLV Format

Where:

- o Type: 2 octet field with value TBD in case of IS-IS and TBD in case of OSPFv3, see Section 8.
- o Length: 2 octet field with the total length of the value portion of the TLV.
- o Function Code: 2 octet field. The Endpoint Function code point for this SRv6 SID as defined in [I-D.filsfils-spring-srv6-network-programming].
- o Flags: 1 octet of flags with the following definition:

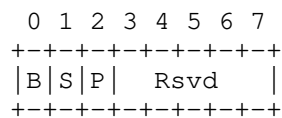


Figure 7: SRv6 LAN End.X SID TLV Flags Format

- * B-Flag: Backup Flag. If set, the SID is eligible for protection (e.g. using IPFRR) as described in [RFC8355].
- * S-Flag: Set Flag. When set, the S-Flag indicates that the SID refers to a set of adjacencies (and therefore MAY be assigned to other adjacencies as well).
- * P-Flag: Persistent Flag: When set, the P-Flag indicates that the SID is persistently allocated, i.e., the value remains consistent across router restart and/or interface flap.
- * Rsvd bits: Reserved for future use and MUST be zero when originated and ignored when received.
- o Algorithm: 1 octet field. Algorithm associated with the SID. Algorithm values are defined in the IGP Algorithm Type registry.
- o Weight: 1 octet field. The value represents the weight of the SID for the purpose of load balancing. The use of the weight is defined in [RFC8402].
- o Reserved: 1 octet field that SHOULD be set to 0 and MUST be ignored on receipt.
- o Neighbor ID : 6 octets of ISIS System ID of the neighbor for the ISIS SRv6 LAN End.X SID TLV and 4 octets of OSPFv3 Router-id of the neighbor for the OSPFv3 SRv6 LAN End.X SID TLV.
- o SID: 16 octet field. This field encodes the advertised SRv6 SID as 128 bit value.
- o Sub-TLVs : currently none defined. Used to advertise sub-TLVs that provide additional attributes for the given SRv6 LAN End.X SID.

4.3. SRv6 Link MSD Types

The Link MSD TLV [I-D.ietf-idr-bgp-ls-segment-routing-msd] of the BGP-LS Attribute of the Link NLRI is also used to advertise the limits and the supported Segment Routing Header (SRH) operations supported on the specific link by the SRv6 capable node. The SRv6 MSD Types specified in [I-D.bashandy-isis-srv6-extensions] are also used with the BGP-LS Link MSD TLV as these codepoints are shared between IS-IS, OSPF and BGP-LS protocols. The description and semantics of these new MSD types for BGP-LS are identical as specified [I-D.bashandy-isis-srv6-extensions] and summarized in the table below:

| MSD Type | Description |
|----------|-----------------------|
| TBD | Maximum Segments Left |
| TBD | Maximum End Pop |
| TBD | Maximum T.Insert |
| TBD | Maximum T.Encaps |
| TBD | Maximum End D |

Figure 8: SRv6 Link MSD Types

Each MSD type is encoded as a one octet type followed by a one octet value.

5. SRv6 Prefix Attributes

SRv6 attributes with an IPv6 prefix are advertised using the new BGP-LS Attribute TLVs defined in this section and associated with the BGP-LS Prefix NLRI.

5.1. SRv6 Locator TLV

As described in [I-D.filsfils-spring-srv6-network-programming], an SRv6 SID is 128 bits and represented as

LOC:FUNCT

where LOC (the locator portion) is the L most significant bits and FUNCT is the 128-L least significant bits. L is called the locator length and is flexible. A node is provisioned with one or more locators supported by that node. Locators are covering prefixes for the set of SIDs provisioned on that node. These Locators are advertised as BGP-LS Prefix NLRI objects along with the SRv6 Locator TLV in its BGP-LS Attribute.

The IPv6 Prefix matching the Locator MAY be also advertised as a prefix reachability by the underlying routing protocol. In this case, the Prefix NLRI would be also associated with the Prefix Metric TLV that carries the routing metric for this prefix. When the Locator prefix is not being advertised as a prefix reachability, then the Prefix NLRI would have the SRv6 Locator TLV associated with it but no Prefix Metric TLV. In the absence of Prefix Metric TLV, the consumer of the BGP-LS topology information MUST NOT interpret the Locator prefix as a prefix reachability routing advertisement.

The SRv6 Locator TLV has the following format:

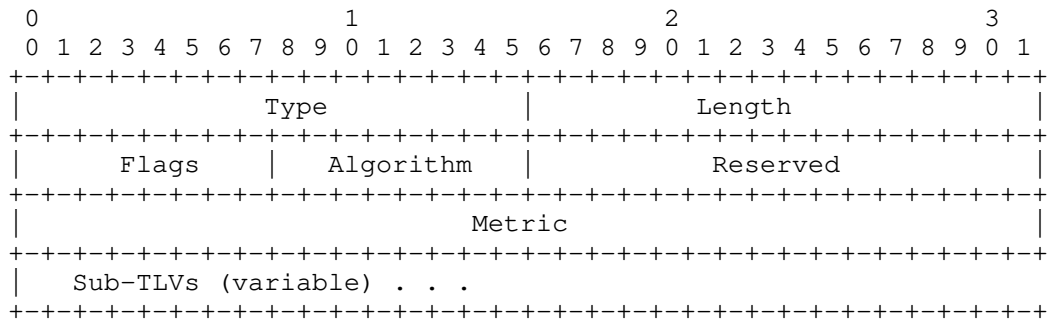


Figure 9: SRv6 Locator TLV Format

Where:

Type: 2 octet field with value TBD, see Section 8.

Length: 2 octet field with the total length of the value portion of the TLV.

Flags: 1 octet of flags with the following definition:

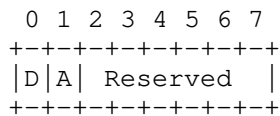


Figure 10: SRv6 Locator TLV Flags Format

- * D-Flag: Indicates that the locator has been leaked into the IGP domain when set. IS-IS operations for this are discussed in [I-D.bashandy-isis-srv6-extensions].
- * A-Flag: When the Locator is associated with anycast destinations, the A flag SHOULD be set. Otherwise, this bit MUST be clear.
- * Reserved bits: Reserved for future use and MUST be zero when originated and ignored when received.

Algorithm: 1 octet field. Algorithm associated with the SID. Algorithm values are defined in the IGP Algorithm Type registry.

Reserved: 2 octet field. The value MUST be zero when originated and ignored when received.

Metric: 4 octet field. The value of the metric for the Locator.

Sub-TLVs : currently none defined. Used to advertise sub-TLVs that provide additional attributes for the given SRv6 Locator.

6. SRv6 SID NLRI

SRv6 SID information is advertised in BGP UPDATE messages using the MP_REACH_NLRI and MP_UNREACH_NLRI attributes [RFC4760]. The "Link-State NLRI" defined in [RFC7752] is extended to carry the SRv6 SID information.

A new "Link-State NLRI Type" is defined for SRv6 SID information as following:

- o Link-State NLRI Type: SRv6 SID NLRI (value TBD see IANA Considerations Section 8.1).

The format of this new NLRI type is as shown in the following figure:

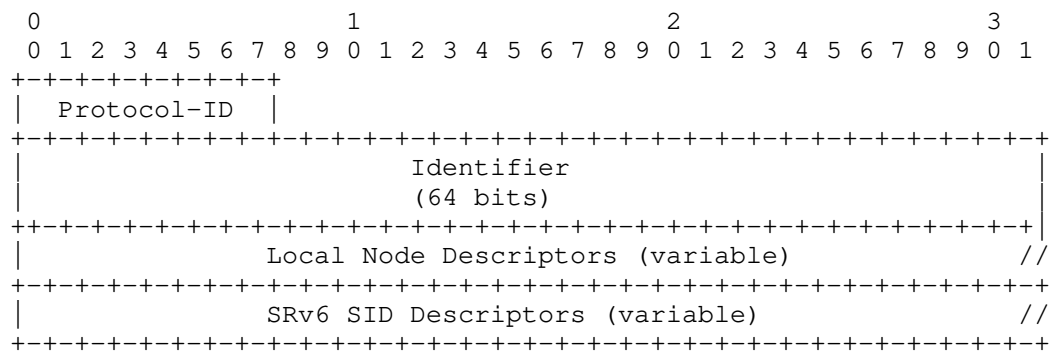


Figure 11: SRv6 SID NLRI Format

Where:

- o Protocol-ID: 1 octet field that specifies the protocol component through which BGP-LS learns the SRv6 SIDs of the node. The following Protocol-IDs apply to the SRv6 SID NLRI:

| Protocol-ID | NLRI information source protocol |
|-------------|----------------------------------|
| 1 | IS-IS Level 1 |
| 2 | IS-IS Level 2 |
| 4 | Direct |
| 5 | Static configuration |
| 6 | OSPFv3 |
| 7 | BGP |

Figure 12: Protocol IDs for SRv6 SID NLRI

- o Identifier: 8 octet value as defined in [RFC7752].
- o Local Node Descriptors TLV: as defined in [RFC7752] for IGPs, local and static configuration and as defined in [I-D.ietf-idr-bgpls-segment-routing-epe] for BGP protocol.
- o SRv6 SID Descriptors: MUST include the SRv6 SID Information TLV defined in Section 6.1 and optionally MAY include the Multi-Topology Identifier TLV as defined in [RFC7752].

New TLVs carried in the BGP Link State Attribute defined in [RFC7752] are also defined in order to carry the attributes of a SRv6 SID in Section 7.

6.1. SRv6 SID Information TLV

A SRv6 SID is a 128 bit value [I-D.filsfils-spring-srv6-network-programming] and is encoded using the SRv6 SID Information TLV.

The TLV has the following format:

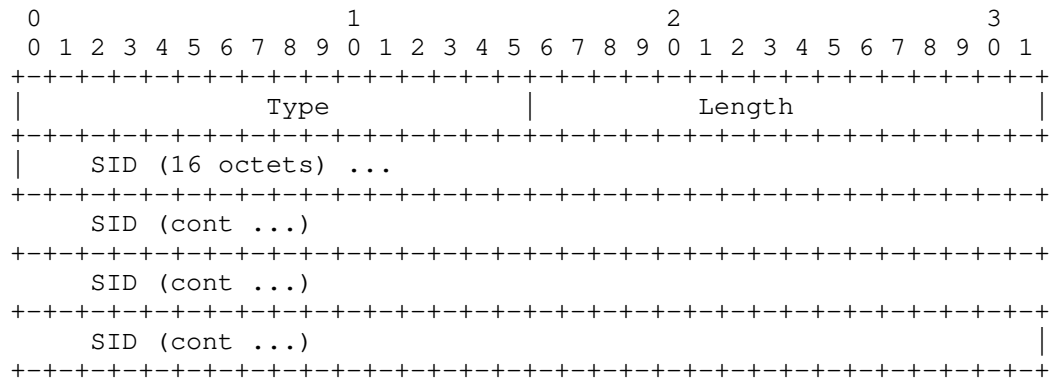


Figure 13: SRv6 SID Information TLV Format

Where:

Type: 2 octet field with value TBD, see Section 8.

Length: 2 octet field with value set to 16.

SID: 16 octet field. This field encodes the advertised SRv6 SID as 128 bit value.

7. SRv6 SID Attributes

This section specifies the new TLVs to be carried in the BGP Link State Attribute associated with the BGP-LS SRv6 SID NLRI.

7.1. SRv6 Endpoint Function TLV

Each SRv6 SID instantiated in the "My SID Table" of an SRv6 capable node has a specific instruction bound to it. A set of well-known functions that can be associated with a SID are defined in [I-D.filsfils-spring-srv6-network-programming].

The SRv6 Endpoint Function TLV is a mandatory TLV that MUST be included in the BGP-LS Attribute associated with the BGP-LS SRv6 SID NLRI. The TLV has the following format:

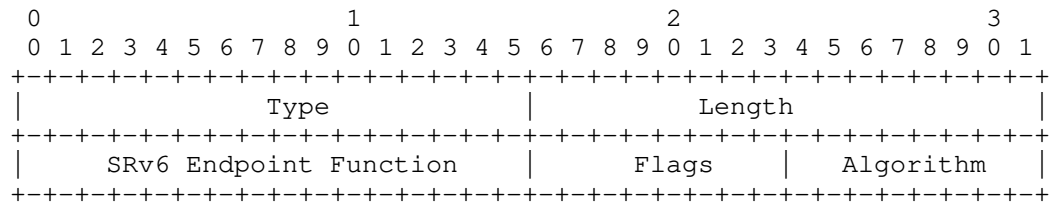


Figure 14: SRv6 Endpoint Function TLV

Where:

Type: 2 octet field with value TBD, see Section 8.

Length: 2 octet field with the value 4.

Function Code: 2 octet field. The Endpoint Function code point for this SRv6 SID as defined in [I-D.filsfils-spring-srv6-network-programming].

Flags: 1 octet of flags with the none defined currently. Reserved for future use and MUST be zero when originated and ignored when received.

Algorithm: 1 octet field. Algorithm associated with the SID. Algorithm values are defined in the IGP Algorithm Type registry.

7.2. SRv6 BGP Peer Node SID TLV

The BGP Peer Node SID and Peer Set SID for SR with MPLS dataplane are specified in [I-D.ietf-idr-bgpls-segment-routing-epe]. The similar Peer Node and Peer Set SID functionality can be realized with SRv6 using the END.X SRv6 SID. The SRv6 BGP Peer Node SID TLV is an optional TLV for use in the BGP-LS Attribute for an SRv6 SID NLRI corresponding to BGP protocol. This TLV MUST be included along with SRv6 End.X SID that is associated with the BGP Peer Node or Peer Set functionality.

The TLV has the following format:

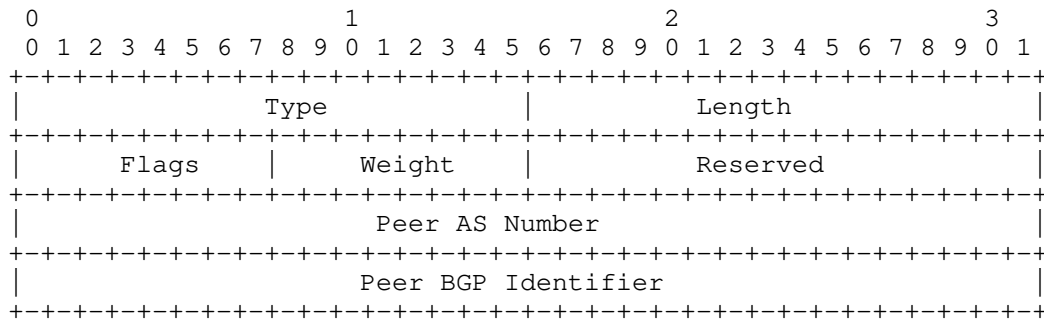


Figure 15: SRv6 BGP Peer Node SID TLV Format

Where:

- o Type: 2 octet field with value TBD, see Section 8.
- o Length: 2 octet field with the value 12.
- o Flags: 1 octet of flags with the following definition:

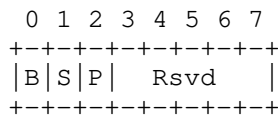


Figure 16: SRv6 BGP Peer End.X SID TLV Flags Format

- * B-Flag: Backup Flag. If set, the SID is eligible for protection (e.g. using IPFRR) as described in [RFC8355].
- * S-Flag: Set Flag. When set, the S-Flag indicates that the SID refers to a set of BGP peering sessions (i.e. BGP Peer Set SID functionality) and therefore MAY be assigned to one or more End.X SIDs associated with BGP peer sessions.
- * P-Flag: Persistent Flag: When set, the P-Flag indicates that the SID is persistently allocated, i.e., the value remains consistent across router restart and/or session flap.
- * Rsvd bits: Reserved for future use and MUST be zero when originated and ignored when received.

- o Weight: 1 octet field. The value represents the weight of the SID for the purpose of load balancing. The use of the weight is defined in [RFC8402].
- o Peer AS Number : 4 octets of BGP AS number of the peer router.
- o Peer BGP Identifier : 4 octets of the BGP Identifier (BGP Router-ID) of the peer router.

For a SRv6 BGP EPE Peer Node SID, one instance of this TLV is associated with the SRv6 SID. For SRv6 BGP EPE Peer Set SID, multiple instances of this TLV (one for each peer in the "peer set") are associated with the SRv6 SID and the S (set/group) flag is SET.

8. IANA Considerations

This document requests assigning code-points from the IANA "Border Gateway Protocol - Link State (BGP-LS) Parameters" registry as described in the sub-sections below.

8.1. BGP-LS NLRI-Types

The following codepoints is suggested (to be assigned by IANA) from within the sub-registry called "BGP-LS NLRI-Types":

| Type | NLRI Type | Reference |
|------|-----------|---------------|
| 6 | SRv6 SID | this document |

Figure 17: SRv6 SID NLRI Type Codepoint

8.2. BGP-LS TLVs

The following TLV codepoints are suggested (to be assigned by IANA) from within the sub-registry called "BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs":

| TLV Code Point | Description | Value defined in |
|----------------|-------------------------------|------------------|
| TBD | SRv6 Capabilities TLV | this document |
| TBD | SRv6 End.X SID TLV | this document |
| TBD | IS-IS SRv6 LAN End.X SID TLV | this document |
| TBD | OSPFv3 SRv6 LAN End.X SID TLV | this document |
| TBD | SRv6 Locator TLV | this document |
| TBD | SRv6 SID Information TLV | this document |
| TBD | SRv6 Endpoint Function TLV | this document |
| TBD | SRv6 BGP Peer Node SID TLV | this document |

Figure 18: SRv6 BGP-LS Attribute TLV Codepoints

9. Manageability Considerations

This section is structured as recommended in[RFC5706]

10. Operational Considerations

10.1. Operations

Existing BGP and BGP-LS operational procedures apply. No additional operation procedures are defined in this document.

11. Security Considerations

Procedures and protocol extensions defined in this document do not affect the BGP security model. 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.

12. Contributors

Arjun Sreekantiah
Individual
US

Ies Ginsberg
Cisco Systems
US
Email: ginsberg@cisco.com

Shunwan Zhuang
Huawei
China
Email: zhuangshunwan@huawei.com

13. Acknowledgements

The authors would like to thank Peter Psenak and Arun Babu for their review of this document and their comments.

14. References

14.1. Normative References

[I-D.ali-spring-srv6-oam]

Ali, Z., Filsfils, C., Kumar, N., Pignataro, C., faiqbal@cisco.com, f., Gandhi, R., Leddy, J., Matsushima, S., Raszuk, R., daniel.voyer@bell.ca, d., Dawra, G., Peirens, B., Chen, M., and G. Naik, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", draft-ali-spring-srv6-oam-02 (work in progress), October 2018.

[I-D.bashandy-isis-srv6-extensions]

Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Routing over IPv6 Dataplane", draft-bashandy-isis-srv6-extensions-05 (work in progress), March 2019.

[I-D.dawra-idr-srv6-vpn]

Dawra, G., Filsfils, C., Dukes, D., Brissette, P., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Decraene, B., Matsushima, S., and S. Zhuang, "BGP Signaling for SRv6 based Services.", draft-dawra-idr-srv6-vpn-05 (work in progress), October 2018.

[I-D.filsfils-spring-srv6-network-programming]

Filsfils, C., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming-07 (work in progress), February 2019.

[I-D.ietf-6man-segment-routing-header]

Filsfils, C., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-16 (work in progress), February 2019.

- [I-D.ietf-idr-bgp-ls-segment-routing-ext]
Previdi, S., Talaulikar, K., Filsfils, C., Gredler, H.,
and M. Chen, "BGP Link-State extensions for Segment
Routing", draft-ietf-idr-bgp-ls-segment-routing-ext-12
(work in progress), March 2019.
- [I-D.ietf-idr-bgp-ls-segment-routing-msd]
Tantsura, J., Chunduri, U., Mirsky, G., Sivabalan, S., and
N. Triantafyllis, "Signaling MSD (Maximum SID Depth) using
Border Gateway Protocol Link-State", draft-ietf-idr-bgp-
ls-segment-routing-msd-04 (work in progress), February
2019.
- [I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray,
S., and J. Dong, "BGP-LS extensions for Segment Routing
BGP Egress Peer Engineering", draft-ietf-idr-bgpls-
segment-routing-epe-17 (work in progress), October 2018.
- [I-D.li-ospf-ospfv3-srv6-extensions]
Li, Z., Hu, Z., Cheng, D., Talaulikar, K., and P. Psenak,
"OSPFv3 Extensions for SRv6", draft-li-ospf-
ospfv3-srv6-extensions-03 (work in progress), March 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>>.
- [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>>.
- [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>>.

14.2. Informative References

- [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>>.
- [RFC4272] Murphy, S., "BGP Security Vulnerabilities Analysis", RFC 4272, DOI 10.17487/RFC4272, January 2006, <<https://www.rfc-editor.org/info/rfc4272>>.
- [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>>.
- [RFC5706] Harrington, D., "Guidelines for Considering Operations and Management of New Protocols and Protocol Extensions", RFC 5706, DOI 10.17487/RFC5706, November 2009, <<https://www.rfc-editor.org/info/rfc5706>>.
- [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>>.
- [RFC8355] Filsfils, C., Ed., Previdi, S., Ed., Decraene, B., and R. Shakir, "Resiliency Use Cases in Source Packet Routing in Networking (SPRING) Networks", RFC 8355, DOI 10.17487/RFC8355, March 2018, <<https://www.rfc-editor.org/info/rfc8355>>.

Authors' Addresses

Gaurav Dawra (editor)
LinkedIn
USA

Email: gdawra.ietf@gmail.com

Clarence Filsfils
Cisco Systems
Belgium

Email: cfilsfil@cisco.com

Ketan Talaulikar (editor)
Cisco Systems
India

Email: ketant@cisco.com

Mach Chen
Huawei
China

Email: mach.chen@huawei.com

Daniel Bernier
Bell Canada
Canada

Email: daniel.bernier@bell.ca

Jim Uttaro
AT&T
USA

Email: jul738@att.com

Bruno Decraene
Orange
France

Email: bruno.decraene@orange.com

Hani Elmalky
Ericsson
USA

Email: hani.elmalky@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).

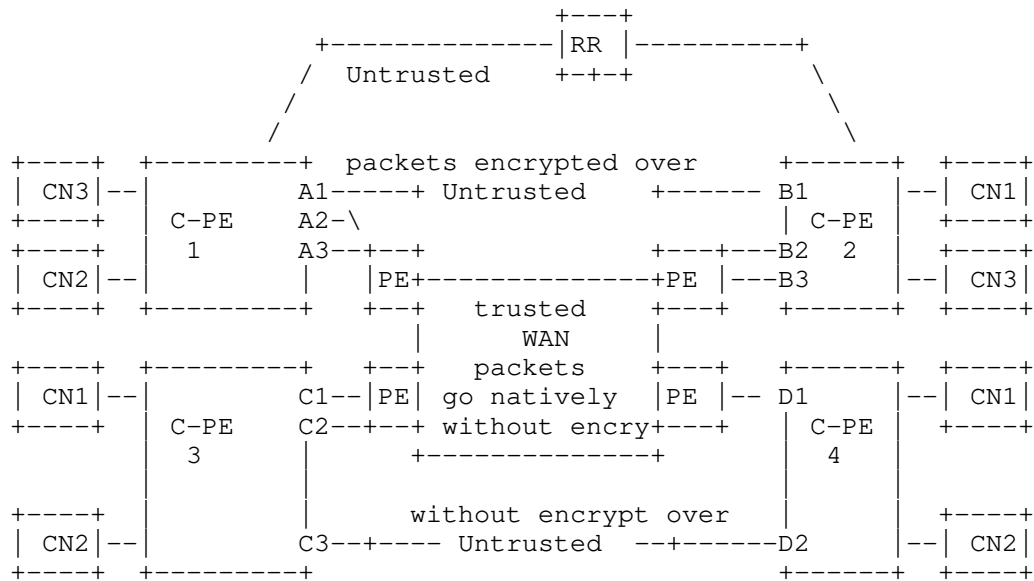
OnPrem: On Premises data centers and branch offices

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

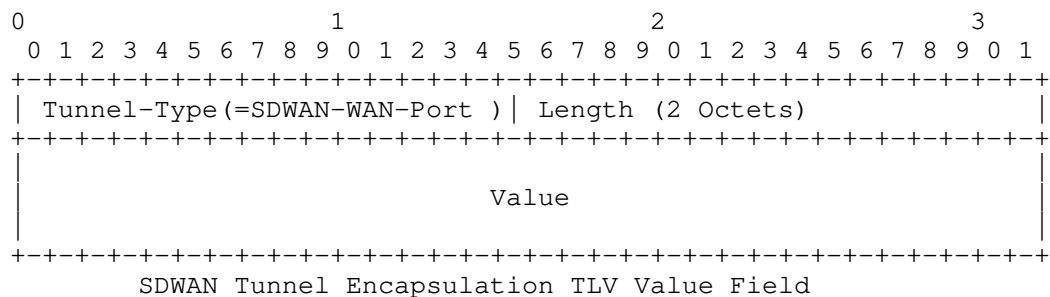
The content of the SDWAN Port properties is encoded in the Tunnel Encapsulation Attribute defined in [Tunnel-Encap] using a new Tunnel-Type TLV (code point to be assigned by IANA from the "BGP Tunnel Encapsulation Attribute Tunnel Types" registry).

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:



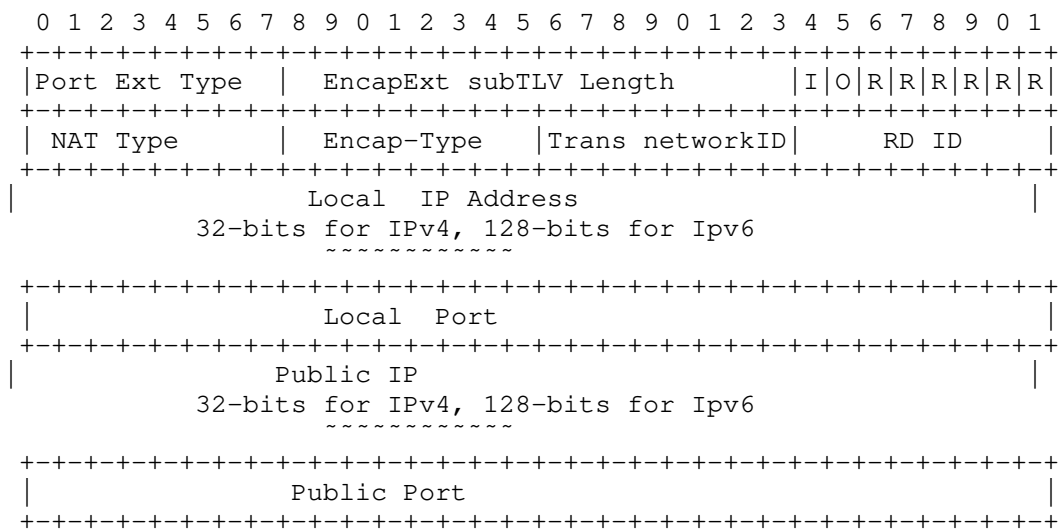
Where:

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

SPRING
Internet-Draft
Intended status: Standards Track
Expires: December 4, 2020

S. Hegde
S. Sangli
M. Srivastava
Juniper Networks Inc.
X. Xu
Alibaba Inc.
June 2, 2020

BGP-LS Extensions for Inter-AS TE using EPE based mechanisms
draft-hegde-idr-bgp-ls-epe-inter-as-03

Abstract

In certain network deployments, a single operator has multiple Autonomous Systems(AS) to facilitate ease of management. A multiple AS network design could also be a result of network mergers and acquisitions. In such scenarios, a centralized Inter-domain TE approach could provide most optimal allocation of resources and the most controlled path placement. BGP-LS-EPE [I-D.ietf-idr-bgp-ls-segment-routing-epe] describes an extension to BGP Link State (BGP-LS) for the advertisement of BGP Peering Segments along with their BGP peering node and inter-AS link information, so that efficient BGP Egress Peer Engineering (EPE) policies and strategies can be computed based on Segment Routing. This document describes extensions to the BGP-LS EPE to enable it to be used for inter-AS Traffic-Engineering (TE) purposes.

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 December 4, 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. Reference Topology | 3 |
| 3. Fast Reroute Label | 4 |
| 4. TE Link attributes of PeerNode SID | 5 |
| 5. TE Link attributes of PeerAdj SID | 5 |
| 6. Link address TLV | 6 |
| 7. Example Advertisements | 7 |
| 8. Backward Compatibility | 9 |
| 9. Security Considerations | 9 |
| 10. IANA Considerations | 9 |
| 11. Acknowledgements | 9 |
| 12. References | 10 |
| 12.1. Normative References | 10 |
| 12.2. Informative References | 10 |
| Authors' Addresses | 10 |

1. Introduction

Segment Routing (SR) leverages source routing. A node steers a packet through a controlled set of instructions, called segments, by prepending the packet with an SR header with segment identifiers (SID). A SID can represent any instruction, topological or service-based. SR segments allows to enforce a flow through any topological path or service function while maintaining per-flow state only at the ingress node of the SR domain.

As there is no per-path state in the network, the bandwidth management for the paths is expected to be handled by a centralized entity which has a complete view of:

1. Up-to-date topology of the network
2. Resources, States and Attributes of links and nodes of the network
3. Run-time utilization/availability of resources

The BGP Link-State extensions provide mechanisms whereby link-state and TE information can be propagated in a network and a consumer of such BGP LS updates may build topology, provide bandwidth calendaring and other traffic engineering services. The centralized entity can be such a consumer (also referred to as controller). In the case of multi-AS networks, the controller needs to learn the per-AS network information and the inter-AS link information thus arriving at a consolidated Traffic Engineering Database which can be used to compute end-to-end Traffic Engineering Path. The controller can learn the topology, link-state and TE information from each of the AS networks either by participating in their IGPs or by listening to BGP LS updates [RFC7752]. Similar information about the inter-AS links can be learnt via BGP-LS EPE [I-D.ietf-idr-bgppls-segment-routing-epe] along with extensions defined in this document.

2. Reference Topology

The controller learns TE attributes of all the links, including the inter-AS links and uses the attributes to compute constrained paths. The controller should be able to correlate the inter-AS links for bidirectional connectivity from both ASes.

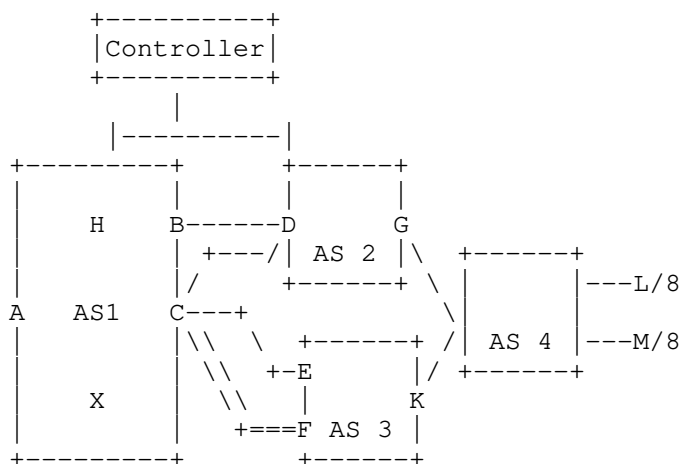


Figure 1: Reference Diagram

The reference diagram from [I-D.ietf-spring-segment-routing-central-epe] represents multiple Autonomous Systems connected to each other. When the Multiple ASes belong to same operator and are organised into separate domains for operational purposes, it is advantageous to support Traffic-Engineering across the ASes including the inter-AS links. The controller has visibility of all of the ASes by means of IGP topology exported via BGP-LS [RFC7752], or other means. In addition, the inter-AS links and the labels associated with the inter-AS links are exported via [I-D.ietf-idr-bgppls-segment-routing-epe]. The controller needs to correlate the information acquired from all of the ASes, including the inter-AS links in order to get a view of the unified topology so that it can build end-to-end Traffic-Engineered paths.

3. Fast Reroute Label

[I-D.ietf-spring-segment-routing-central-epe] section 3.6 describes mechanisms to provide Fast Reroute (FRR) protection for the EPE Labels. The BGP-LS EPE [I-D.ietf-idr-bgppls-segment-routing-epe] describes "B" bit to indicate that a PeerNodeSID or PeerAdjSID is eligible for backup. However, it does not specify what is the behaviour when the failure kicks-in. The controller needs to know which links are used for protection so that admission control and failure simulation can be done effectively and appropriate inter-AS links used for path construction.

This document defines a new flag "F" in the Peering SIDs TLV to indicate a SID as an FRR SID. With the "F" flag set, the protection for any peering SID can be specified using another PeerAdjSID, PeerNodeSID or PeerSetSID to the controller. If the protection is achieved by fallback to local IP lookup, FRR SID SHOULD not be advertised. The link(s) represented by the FRR SID will carry the traffic when there is a failure. These SIDs are included as an FRR SIDs in the peerAdjSID, PeerNodeSID and PeerSetSID advertisements.

```

      0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|V|L|B|P|F|Rsvd|
+---+---+---+---+---+---+

```

Figure 2: Peering SID TLV Flags Format

* F-Flag: FRR Label Flag: If set, the peer SID where the FRR Label appears is using backup links represented by FRR Label.

4. TE Link attributes of PeerNode SID

In any eBGP deployment, the peering session can be either single-hop or multi-hop. For single-hop eBGP sessions, the peering address is that of the directly attached interface to which the session is pinned down. For multi-hop eBGP session, the peering address is reachable over more than one interface and that the peering session is not pinned down to any of the directly attached interfaces.

A Peer Node Segment is a segment describing a peer, including the SID (PeerNodeSID) allocated to it. The link descriptors for the PeerNodeSID include the addresses used by the BGP session encoded using TLVs as defined in [RFC7752]. Since the eBGP session can be either single-hop or multi-hop, the IP address used by BGP session as local/neighbour is not sufficient to identify the underlaying interface(s). Also, the controller needs to know the links associated with the PeerNodeSID, to be able derive TE link attributes. This can be achieved by including the interface local and remote addresses in the Link attributes in PeerNodeSID NLRI. This document defines a new link attribute TLV name Interface Address TLV. PeerNodeSID NLRI MAY optionally include Interface Address TLV.

5. TE Link attributes of PeerAdj SID

PeerAdjSID MUST be advertised for each inter-AS link for the purposes of inter-AS TE. The PeerAdjSID should contain link TE attributes such as bandwidth, admin-group etc. The PeerAdjSID should also

contain the local and remote interface IPv4/IPv6 addresses which is used for correlating the links. PeerNodeSID SHOULD contain the additional attribute of link local address which is used by the controller to find corresponding PeerAdjSID and hence the corresponding link TE attributes.

A peerAdj segment carries mandatory link descriptors as local and remote link id. Remote link id of the neighboring ASBR is not readily available. [I-D.ietf-idr-bgppls-segment-routing-epe] suggests to carry the value '0' for the remote link id. The Controller needs to associate the links in both directions to effectively handle failure notifications and for this purpose a unique remote link is necessary. The remote link ID cannot be manually configured on the router as the link-ids generally change over router reboot etc and hence manual configuration is operationally very difficult to manage. This document mandates advertisement of local and remote interface addresses for the inetr-AS TE purposes.

The Unnumbered interface is not in the scope of this document.

6. Link address TLV

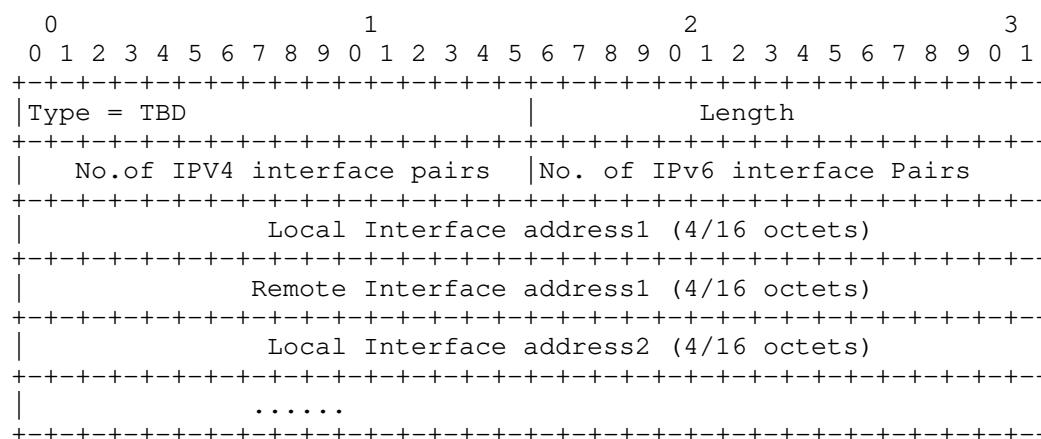


Figure 3: Link Address TLV carried as attribute

Type : TBD

Length : variable based on ipv4/ipv6 interface address

Number of IPv4 interface pairs:

Number of IPv6 interface pairs:

There may be a number of parallel interfaces and few or all of them may be used for the PeerNodeSID. These interfaces may have both IPV4 and IPV6 address or some interfaces may be IPv4 only and some IPv6 only. The total number of IPv4 and IPv6 interface address count is carried separately in above fields.

Local Interface Address :

The interface local address ipv4/ipv6 which corresponds to the PeerNodeSID MUST be specified. For IPv4, this field is 4 octets; for IPv6, this field is 16 octets.

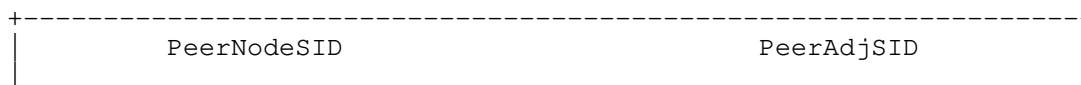
Remote Interface Address :

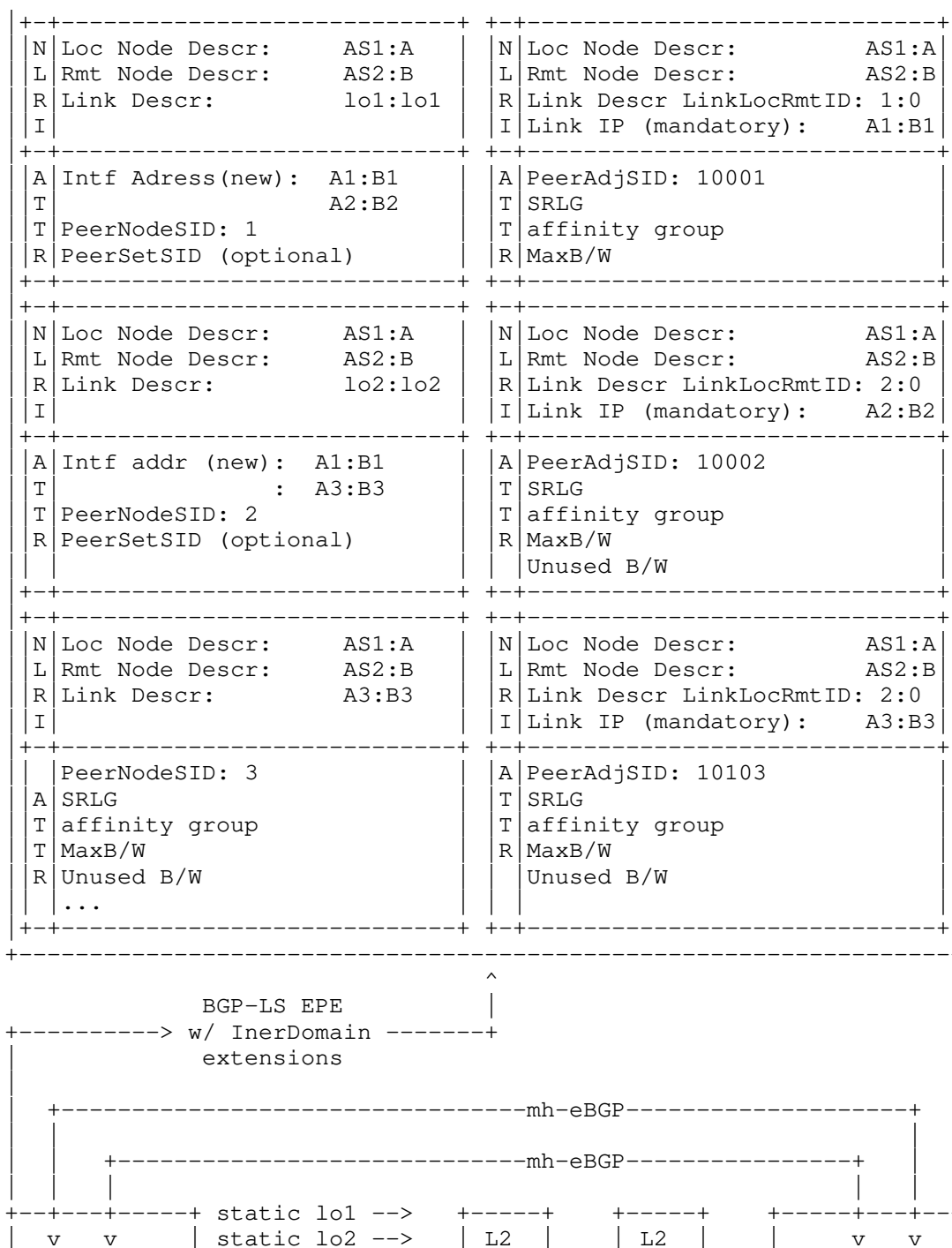
The interface remote address ipv4/ipv6 which corresponds to the PeerNodeSID MUST be specified. For IPv4, this field is 4 octets; for IPv6, this field is 16 octets.

There can be multiple Layer 3 interfaces on which a peerNodeSID loadbalances the traffic. All such interfaces local/remote address MUST be included when a Link address TLV is added. When a single Layer 3 interface consists of multiple addresses or when a link has both IPv4 and IPv6 addresses configured, It is sufficient to include one such pair (either IPV4 or IPV6) address for the PeerNodeSID advertisement. When a PeerNodeSID load-balances over few interfaces with IPv4 only address and few interfaces with IPv6 address then the Link address TLV should list all IPv4 address pairs together followed by IPv6 address pairs.

7. Example Advertisements

The below diagram represents two ASBR routers and inter-AS links between them. The inter-AS links could be connected via switches L1 and L2 as shown in the diagram or via Point-to-point links A2->B2, A3->B3 as shown in the diagram below. In the below example, lets assume peerNodeSID 1 is configured to use peerAdjSID 10002 then PeerNodeSID 1 will have the B bit set which means the PeerNodeSID 1 is eligible for backup. Label 10002 is added to the PeerNodeSID with a "F" bit set, which means 10002 is a backup for PeerNodeSID 1.





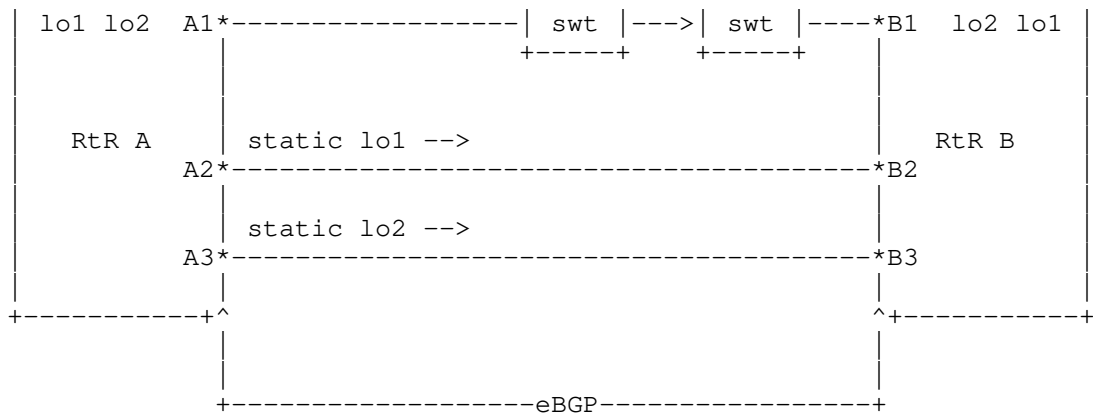


Figure 4: Example Advertisements

8. Backward Compatibility

The extension proposed in this document is backward compatible with procedures described in [I-D.ietf-idr-bgppls-segment-routing-epe] and [I-D.ietf-spring-segment-routing-central-epe]

9. Security Considerations

TBD

10. IANA Considerations

New attribute TLV in BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs registry

| TLV Code Point | Description | IS-IS TLV /Sub-TLV | Reference (RFC/Section) |
|----------------|------------------|--------------------|-------------------------|
| TBD | Link address TLV | NA | This draft |

Figure 5: IANA code point

11. Acknowledgements

Thanks to Julian Lucek and Rafal Jan Szarecki for careful review and suggestions.

12. References

12.1. Normative References

- [I-D.ietf-idr-bgppls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgppls-segment-routing-epe-19 (work in progress), May 2019.
- [I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.
- [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>>.

12.2. Informative References

- [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

Shraddha Hegde
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: shraddha@juniper.net

Srihari Sangli
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: ssangli@juniper.net

Mukul Srivastava
Juniper Networks Inc.

Email: msri@juniper.net

Xiaohu Xu
Alibaba Inc.
Beijing
China

Email: xiaohu.xxh@alibaba-inc.com

IDR
Internet-Draft
Intended status: Standards Track
Expires: September 26, 2019

J. Heitz
P. Narasimha
S. Gulrajani
Cisco
March 25, 2019

BGP Diagnostic Path Attribute
draft-heitz-idr-diagnostic-attr-01

Abstract

A BGP path attribute for the purpose of network diagnostics is described. It is non-transitive, such that BGP speakers will not forward it by default. It is structured as a list of elements. An element begins with the BGP identifier and AS number of the speaker that appends the element and includes a list of TLVs. Any speaker can add or remove an element to/from the list. TLVs for a timestamp and for a checksum are described.

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 September 26, 2019.

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

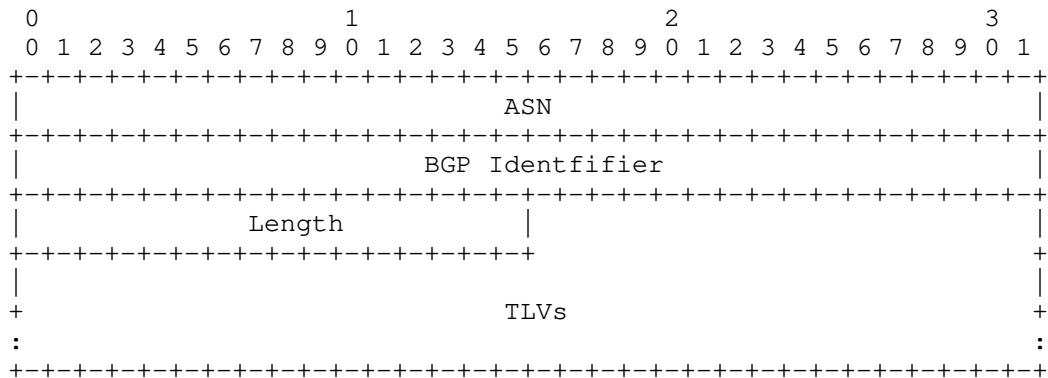
| | |
|---|---|
| 1. Introduction | 2 |
| 2. Data Formats | 2 |
| 2.1. Checksum TLV | 3 |
| 2.2. Timestamp TLV | 4 |
| 3. Usage | 5 |
| 4. Operational Considerations | 5 |
| 5. Error Handling | 6 |
| 6. Security Considerations | 6 |
| 7. IANA Considerations | 6 |
| 8. Acknowledgements | 7 |
| 9. Normative References | 7 |
| Authors' Addresses | 8 |

1. Introduction

A BGP path attribute for the purpose of network diagnostics is described. It is non-transitive, such that BGP speakers that do not recognize the attribute will not propagate it by default. Even speakers that do recognize the attribute MUST NOT propagate it by default. A speaker MAY propagate the attribute if it is configured to do so and MAY add it's own information as it does so. The attribute is structured as a list of elements. An element begins with the BGP identifier and AS number of the speaker that appends the element and includes a list of TLVs. Any speaker can append or remove an element to/from the list. TLVs for a timestamp and for a checksum are described. The diagnostic attribute may be sent in a withdraw message.

2. Data Formats

The BGP diagnostic consists of a series of elements, each of which is formatted as follows.

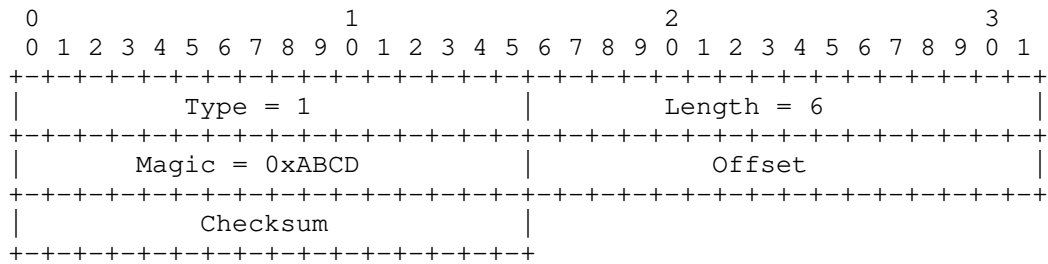


The fields are as follows:

- ASN - 4 octet Autonomous System Number of the speaker that appended this element.
- BGP Identifier - BGP Identifier of the speaker that appended this element.
- Length - The number of octets comprising the TLVs of this element. If there were no TLVs included, this length would be 0.
- TLVs - Any number of TLVs as further described. Each TLV is optional. Each TLV comprises 2 octets of Type, then 2 octets specifying the number of octets in the value, then the octets of the value.

2.1. Checksum TLV

A checksum of the BGP message, including the marker field. The checksum is only valid between the sending and receiving speaker. Since a receiving speaker may propagate an update, it will likely change the set of attributes or their order in its own update message, thus making the checksum useless in the propagated update. A BGP speaker MAY remove the checksum TLV from a propagated Diagnostic Path Attribute.

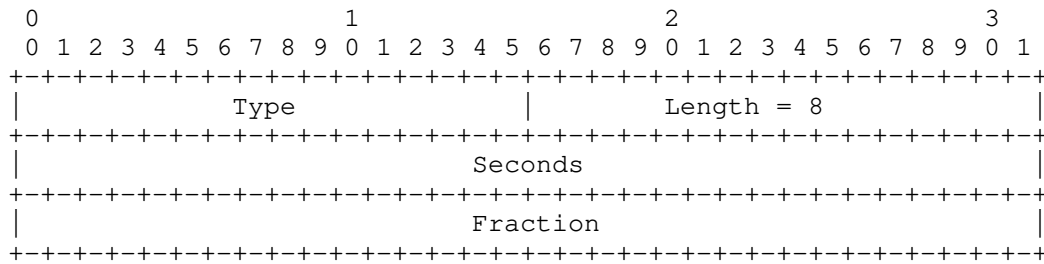


The fields are as follows:

- Type - 1.
- Length - 6.
- Magic - The value 0xABCD. This helps to diagnose corruption when looking at a hexdump.
- Offset - The number of octets from the start of the UPDATE message (start of the marker) to the start of this TLV. This can help to identify corruption due to misaligned segment reassembly.
- Checksum - The 16 bit checksum computed according to [RFC1071].

2.2. Timestamp TLV

The time at which the indicated speaker processed the indicated set of NLRI in the UPDATE message. There are several stages of processing for each NLRI, each of which may be timestamped. These stages vary widely between implementations. Therefore, the type code used for each stage is implementation dependent. One stage is universal. That is the stage when BGP hands the UPDATE message to TCP for transmission. The Type code for the timestamp for that stage is 256. The accuracy of the timestamp depends upon the diagnostic application that requires it and is out of scope of this document. The timestamp has enough bits to describe a time point within a period of 136 years. As time passes, the timestamp will simply wrap from one period to the next. For example, there exist some time points in the year 1900 and the year 2036 with identical timestamps. The determination of the period in which a timestamp occurs is out of scope of this document.



The fields are as follows:

- | | |
|----------|---|
| Type | - 256: The time when BGP hands the UPDATE message off to TCP. |
| | - 257 - 511: Timestamps for any other stages of processing within BGP. The actual values and stages are implementation dependent. |
| Length | - 8. |
| Seconds | - The number of Seconds since 0 h 1 January 1900 UTC as further described in Section 6 of [RFC5905]. |
| Fraction | - A fraction of the above seconds also described in Section 6 of [RFC5905]. |

3. Usage

The Checksum TLV is useful to narrow down a source of corruption of UPDATE messages in each of the software and hardware layers between the actual BGP processes.

Because the Diagnostic Attribute contains a list of speakers that propagated an UPDATE and the attribute can be attached to a withdraw message, it can assist in the diagnosis of route oscillations.

The timestamp TLV is used to narrow down delays in UPDATE processing between BGP speakers and between the various stages of processing within a BGP speaker.

4. Operational Considerations

As with any new BGP attribute, if it is propagated, NLRI packing into BGP UPDATE messages may be affected. This needs to be taken into consideration when using the Timestamp TLV to measure bulk update message timing.

The Diagnostic Path Attribute MAY be sent in an UPDATE message that does not contain an NLRI field [RFC4271] or an MP_REACH_NLRI Path Attribute [RFC4760]. When carried in such a message, it is unlikely to be propagated, although it is possible.

If the addition or extension of the Diagnostic Path Attribute would cause the UPDATE message length to be exceeded, then the attribute SHOULD NOT be added or extended.

If the timestamps among participating speakers are not well synchronized, then the timestamps added by each speaker may appear out of order. In any case, the order in which elements are added to the Diagnostic Attribute can always be determined, because each speaker appends its element to the attribute.

The experimental TLV types may clash. That means that multiple vendors may use the same experimental TLV type code for different purposes unbeknown to each other. To reduce the chance of TLV type code clashes, the type code for an experimental TLV SHOULD be configurable on the speaker. Because the propagation of the Diagnostic Attribute must be configured on each speaker, it is unlikely that two uncoordinated experiments will interfere with each other.

5. Error Handling

A checksum error SHALL NOT be treated as a protocol error. The response is implementation dependent.

A TLV length error SHALL be treated as attribute-discard according to [RFC7606].

An unrecognized TLV MUST not be treated as a protocol error.

6. Security Considerations

This attribute is not forwarded by default. Therefore, it should cause no unexpected ill effects.

7. IANA Considerations

IANA is requested to assign a BGP path attribute value for the BGP Diagnostic Path Attribute.

IANA is requested to create and maintain a registry for the TLV types. The allocation policies as per [RFC8126] are stated for the range of values.

| Range | Allocation Policy |
|-------------|-------------------------|
| 0-32767 | First Come First Served |
| 32768-65535 | Experimental |

| Value | Description | Reference |
|-----------|-------------|-----------|
| 0 | Reserved | This RFC |
| 1 | Checksum | This RFC |
| 256 - 511 | Timestamp | This RFC |

8. Acknowledgements

The authors would like to thank Shyam Sethuram and Bruno Decraene for suggestions that have been added to the document.

9. Normative References

- [RFC1071] Braden, R., Borman, D., and C. Partridge, "Computing the Internet checksum", RFC 1071, DOI 10.17487/RFC1071, September 1988, <<https://www.rfc-editor.org/info/rfc1071>>.
- [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>>.
- [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>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [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>>.

Authors' Addresses

Jakob Heitz
Cisco
170 West Tasman Drive
San Jose, CA, CA 95134
USA

Email: jheitz@cisco.com

Prasad S. Narasimha
Cisco
170 West Tasman Drive
San Jose, CA, CA 95134
USA

Email: snprasad@cisco.com

Sameer Gulrajani
Cisco
170 West Tasman Drive
San Jose, CA, CA 95134
USA

Email: sameerg@cisco.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:

```

+-----+
| list of prefixes (variable) |
+-----+
```

Figure 1: Source Prefix sub-TLV

Each prefix is encoded as following:

```

+-----+
| prefix Length (1 octet) |
+-----+
| Prefix (4 or 16 octets) |
+-----+
```

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;

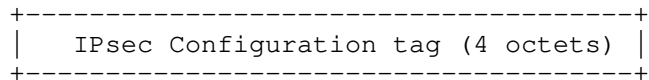


Figure 3: IPsec Configuration Tag

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:

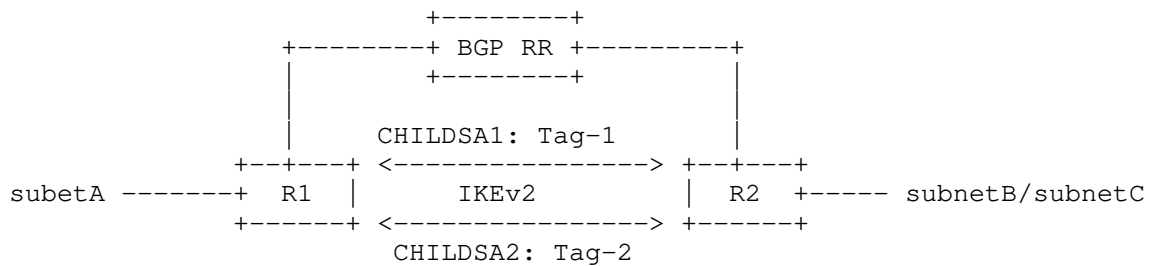


Figure 4: Operation Example

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

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-IGP 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>
            </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>
    </defined-sets>
  </routing-policy>
```

```
<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>T0_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 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
+--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
|   |   |   |   |   |   |   |   |   boolean
|   |   |   |   |   |   |   |   +--ro ineligible-reason?

```

```

        identityref
    +--ro unknown-attributes
        +--ro unknown-attribute*
            [attr-type]
                +--ro optional?          boolean
                +--ro transitive?        boolean
                +--ro partial?            boolean
                +--ro extended?           boolean
                +--ro attr-type           uint8
                +--ro attr-len?           uint16
                +--ro attr-value?         binary
    +--ro reject-reason?                  union
+--ro clear-routes {bt:clear-routes}?
    +---x clear
        +---w input
            +---w clear-at?
                yang:date-and-time
    +--ro output
        +--ro clear-finished-at?
            yang:date-and-time
+--ro adj-rib-in-post
    +--ro routes
        +--ro route* [prefix path-id]
            +--ro prefix
                |
                inet:ipv4-prefix
            +--ro path-id                      uint32
            +--ro attr-index?                  leafref
            +--ro community-index?             leafref
            +--ro ext-community-index?         leafref
            +--ro last-modified?
                |
                yang:timeticks
            +--ro eligible-route?
                |
                boolean
            +--ro ineligible-reason?
                |
                identityref
            +--ro best-path?
                |
                boolean
            +--ro unknown-attributes
                +--ro unknown-attribute*
                    [attr-type]
                        +--ro optional?          boolean
                        +--ro transitive?        boolean
                        +--ro partial?            boolean
                        +--ro extended?           boolean
                        +--ro attr-type           uint8
                        +--ro attr-len?           uint16
                        +--ro attr-value?         binary
            +--ro reject-reason?                  union

```

```

+--ro clear-routes {bt:clear-routes}?
  +---x clear
    +---w input
      +---w clear-at?
        yang:date-and-time
    +--ro output
      +--ro clear-finished-at?
        yang:date-and-time
+--ro adj-rib-out-pre
+--ro routes
  +--ro route* [prefix path-id]
    +--ro prefix
      inet:ipv4-prefix
    +--ro path-id uint32
    +--ro attr-index? leafref
    +--ro community-index? leafref
    +--ro ext-community-index? leafref
    +--ro 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
Intended status: Standards Track
Expires: 3 October 2022

A. Azimov
Qrator Labs & Yandex
E. Bogomazov
Qrator Labs
R. Bush
Internet Initiative Japan & Arrcus, Inc.
K. Patel
Arrcus
K. Sriram
USA NIST
1 April 2022

Route Leak Prevention and Detection using Roles in UPDATE and OPEN
Messages
draft-ietf-idr-bgp-open-policy-24

Abstract

Route leaks are the propagation of BGP prefixes that violate assumptions of BGP topology relationships, e.g., announcing a route learned from one transit provider to another transit provider or a lateral (i.e., non-transit) peer or announcing a route learned from one lateral peer to another lateral peer or a transit provider. These are usually the result of misconfigured or absent BGP route filtering or lack of coordination between autonomous systems (ASes). Existing approaches to leak prevention rely on marking routes by operator configuration, with no check that the configuration corresponds to that of the eBGP neighbor, or enforcement that the two eBGP speakers agree on the peering relationship. This document enhances the BGP OPEN message to establish an agreement of the peering relationship on each eBGP session between autonomous systems in order to enforce appropriate configuration on both sides. Propagated routes are then marked according to the agreed relationship, allowing both prevention and detection of route leaks.

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 3 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 | 3 |
| 2. Terminology | 3 |
| 2.1. Peering Relationships | 4 |
| 3. BGP Role | 5 |
| 3.1. BGP Role Capability | 5 |
| 3.2. Role Correctness | 6 |
| 4. BGP Only to Customer (OTC) Attribute | 8 |
| 5. Additional Considerations | 10 |
| 6. IANA Considerations | 10 |
| 7. Security Considerations | 11 |
| 8. References | 12 |
| 8.1. Normative References | 12 |
| 8.2. Informative References | 13 |
| Acknowledgments | 14 |
| Contributors | 14 |
| Authors' Addresses | 14 |

1. Introduction

Route leaks are the propagation of BGP prefixes that violate assumptions of BGP topology relationships, e.g., announcing a route learned from one transit provider to another transit provider or a lateral (i.e., non-transit) peer or announcing a route learned from one lateral peer to another lateral peer or a transit provider [RFC7908]. These are usually the result of misconfigured or absent BGP route filtering or lack of coordination between autonomous systems (ASes).

Existing approaches to leak prevention rely on marking routes by operator configuration, with no check that the configuration corresponds to that of the eBGP neighbor, or enforcement that the two eBGP speakers agree on the relationship. This document enhances the BGP OPEN message to establish an agreement of the relationship on each eBGP session between autonomous systems in order to enforce appropriate configuration on both sides. Propagated routes are then marked according to the agreed relationship, allowing both prevention and detection of route leaks.

This document specifies a means of replacing the operator-driven configuration-based method of route leak prevention, described above, with an in-band method for route leak prevention and detection.

This method uses a new configuration parameter, BGP Role, which is negotiated using a BGP Role Capability in the OPEN message [RFC5492]. An eBGP speaker may require the use of this capability and confirmation of BGP Role with a neighbor for the BGP OPEN to succeed.

An optional, transitive BGP Path Attribute, called Only to Customer (OTC), is specified in Section 4. It prevents ASes from creating leaks and detects leaks created by the ASes in the middle of an AS path. The main focus/applicability is the Internet (IPv4 and IPv6 unicast route advertisements).

2. Terminology

The terms "local AS" and "remote AS" are used to refer to the two ends of an eBGP session. The "local AS" is the AS where the protocol action being described is to be performed, and "remote AS" is the AS at the other end of the eBGP session in consideration.

The use of the term "route is ineligible" in this document has the same meaning as in [RFC4271], i.e., "route is ineligible to be installed in Loc-RIB and will be excluded from the next phase of route selection."

2.1. Peering Relationships

The terms for peering relationships defined and used in this document (see below) do not necessarily represent business relationships based on payment agreements. These terms are used to represent restrictions on BGP route propagation, sometimes known as the Gao-Rexford model [Gao]. The terms Provider, Customer, and Peer used here are synonymous to the terms "transit provider", "customer", and "lateral (i.e., non-transit) peer", respectively, used in [RFC7908].

The following is a list of BGP Roles for eBGP peering and the corresponding rules for route propagation:

Provider: MAY propagate any available route to a Customer.

Customer: MAY propagate any route learned from a Customer, or locally originated, to a Provider. All other routes MUST NOT be propagated.

Route Server (RS): MAY propagate any available route to a Route Server Client (RS-Client).

Route Server Client (RS-Client): MAY propagate any route learned from a Customer, or locally originated, to an RS. All other routes MUST NOT be propagated.

Peer: MAY propagate any route learned from a Customer, or locally originated, to a Peer. All other routes MUST NOT be propagated.

If the local AS has one of the above Roles (in the order shown), then the corresponding peering relationship with the remote AS is Provider-to-Customer, Customer-to-Provider, RS-to-RS-Client, RS-Client-to-RS, or Peer-to-Peer (i.e., lateral peers), respectively. These are called normal peering relationships.

If the local AS has more than one peering role with the remote AS such peering relation is called Complex. An example is when the peering relationship is Provider-to-Customer for some prefixes while it is Peer-to-Peer for other prefixes [Gao].

A BGP speaker may apply policy to reduce what is announced, and a recipient may apply policy to reduce the set of routes they accept.

Violation of the route propagation rules listed above may result in route leaks [RFC7908]. Automatic enforcement of these rules should significantly reduce route leaks that may otherwise occur due to manual configuration mistakes.

As specified in Section 4, the Only to Customer (OTC) Attribute is used to identify all the routes in the AS that have been received from a Peer, Provider, or RS.

3. BGP Role

The BGP Role characterizes the relationship between the eBGP speakers forming a session. One of the Roles described below SHOULD be configured at the local AS for each eBGP session (see definitions in Section 2) based on the local AS's knowledge of its Role. The only exception is when the eBGP connection is Complex (see Section 5). BGP Roles are mutually confirmed using the BGP Role Capability (described in Section 3.1) on each eBGP session.

Allowed Roles for eBGP sessions are:

- * Provider - the local AS is a transit Provider of the remote AS;
- * Customer - the local AS is a transit Customer of the remote AS;
- * RS - the local AS is a Route Server (usually at an Internet exchange point) and the remote AS is its RS-Client;
- * RS-Client - the local AS is a client of an RS and the RS is the remote AS;
- * Peer - the local and remote ASes are Peers (i.e., have a lateral peering relationship).

3.1. BGP Role Capability

The BGP Role Capability is defined as follows:

- * Code - 9
- * Length - 1 (octet)
- * Value - integer corresponding to speaker's BGP Role (see Table 1).

| Value | Role name (for the local AS) |
|-------|------------------------------|
| 0 | Provider |
| 1 | RS |
| 2 | RS-Client |
| 3 | Customer |
| 4 | Peer (i.e., Lateral Peer) |
| 5-255 | Unassigned |

Table 1: Predefined BGP Role Values

If BGP Role is locally configured, the eBGP speaker MUST advertise BGP Role Capability in the BGP OPEN message. An eBGP speaker MUST NOT advertise multiple versions of the BGP Role Capability. The error handling when multiple BGP Role Capabilities are received is described in Section 3.2.

3.2. Role Correctness

Section 3.1 described how BGP Role encodes the relationship on each eBGP session between autonomous systems (ASes).

The mere receipt of BGP Role Capability does not automatically guarantee the Role agreement between two eBGP neighbors. If the BGP Role Capability is advertised, and one is also received from the peer, the Roles MUST correspond to the relationships in Table 2. If the Roles do not correspond, the BGP speaker MUST reject the connection using the Role Mismatch Notification (code 2, subcode TBD).

| Local AS Role | Remote AS Role |
|---------------|----------------|
| Provider | Customer |
| Customer | Provider |
| RS | RS-Client |
| RS-Client | RS |
| Peer | Peer |

Table 2: Allowed Pairs of Role Capabilities

For backward compatibility, if the BGP Role Capability is sent but one is not received, the BGP Speaker SHOULD ignore the absence of the BGP Role Capability and proceed with session establishment. The locally configured BGP Role is used for the procedures described in Section 4.

An operator may choose to apply a "strict mode" in which the receipt of a BGP Role Capability from the remote AS is required. When operating in the "strict mode", if the BGP Role Capability is sent, but one is not received, then the connection is rejected using the Role Mismatch Notification (code 2, subcode TBD). See comments in Section 7.

If an eBGP speaker receives multiple but identical BGP Role Capabilities with the same value in each, then the speaker considers them to be a single BGP Role Capability and proceeds [RFC5492]. If multiple BGP Role Capabilities are received and not all of them have the same value, then the BGP speaker MUST reject the connection using the Role Mismatch Notification (code 2, subcode TBD).

The BGP Role value for the local AS (in conjunction with the OTC Attribute in the received UPDATE message) is used in the route leak prevention and detection procedures described in Section 4.

4. BGP Only to Customer (OTC) Attribute

The Only to Customer (OTC) Attribute is an optional transitive path attribute of the UPDATE message with Attribute Type Code 35 and a length of 4 octets. The purpose of this attribute is to enforce that once a route is sent to a Customer, Peer, or RS-Client (see definitions in Section 2.1), it will subsequently go only to Customers. The attribute value is an AS number (ASN) determined by the procedures described below.

The following ingress procedure applies to the processing of the OTC Attribute on route receipt:

1. If a route with the OTC Attribute is received from a Customer or RS-Client, then it is a route leak and MUST be considered ineligible (see Section 2).
2. If a route with the OTC Attribute is received from a Peer (i.e., remote AS with a Peer Role) and the Attribute has a value that is not equal to the remote (i.e., Peer's) AS number, then it is a route leak and MUST be considered ineligible.
3. If a route is received from a Provider, Peer, or RS, and the OTC Attribute is not present, then it MUST be added with a value equal to the AS number of the remote AS.

The following egress procedure applies to the processing of the OTC Attribute on route advertisement:

1. If a route is to be advertised to a Customer, Peer, or RS-Client (when the sender is an RS), and the OTC Attribute is not present, then when advertising the route, an OTC Attribute MUST be added with a value equal to the AS number of the local AS.
2. If a route already contains the OTC Attribute, it MUST NOT be propagated to Providers, Peers, or RS(s).

The above-described procedures provide both leak prevention for the local AS and leak detection and mitigation multiple hops away. In the case of prevention at the local AS, the presence of an OTC Attribute indicates to the egress router that the route was learned from a Peer, Provider, or RS, and it can be advertised only to the customers. The same OTC Attribute which is set locally also provides a way to detect route leaks by an AS multiple hops away if a route is received from a Customer, Peer, or RS-Client. For example, if an AS sets the OTC Attribute on a route sent to a Peer and the route is subsequently received by a compliant AS from a Customer, then the receiving AS detects (based on the presence of the OTC Attribute) that the route is a leak.

The OTC Attribute might be set at the egress of the remote AS or at the ingress of the local AS, i.e., if the remote AS is non-compliant with this specification, then the local AS will have to set the OTC Attribute if it is absent. In both scenarios, the OTC value will be the same. This makes the scheme more robust and benefits early adopters.

The OTC Attribute is considered malformed if the length value is not 4. An UPDATE message with a malformed OTC Attribute SHALL be handled using the approach of "treat-as-withdraw" [RFC7606].

The BGP Role negotiation and OTC Attribute based procedures specified in this document are NOT RECOMMENDED to be used between autonomous systems in an AS Confederation [RFC5065]. If an OTC Attribute is added on egress from the AS Confederation, its value MUST equal the AS Confederation Identifier. Also, on egress from the AS Confederation, an UPDATE MUST NOT contain an OTC Attribute with a value corresponding to any Member-AS Number other than the AS Confederation Identifier.

The procedures specified in this document in scenarios that use private AS numbers behind an Internet-facing ASN (e.g., a data center network [RFC7938] or stub customer) may be used, but any details are outside the scope of this document. On egress from the Internet-facing AS, the OTC Attribute MUST NOT contain a value other than the Internet-facing ASN.

Once the OTC Attribute has been set, it MUST be preserved unchanged (this also applies to an AS Confederation).

The described ingress and egress procedures are applicable only for the address families AFI 1 (IPv4) and AFI 2 (IPv6) with SAFI 1 (unicast) in both cases and MUST NOT be applied to other address families by default. The operator MUST NOT have the ability to modify the procedures defined in this section.

5. Additional Considerations

Roles MUST NOT be configured on an eBGP session with a Complex peering relationship. If multiple eBGP sessions can segregate the Complex peering relationship into eBGP sessions with normal peering relationships, BGP Roles SHOULD be used on each of the resulting eBGP sessions.

An operator may want to achieve an equivalent outcome by configuring policies on a per-prefix basis to follow the definitions of peering relations as described in Section 2.1. However, in this case, there are no in-band measures to check the correctness of the per-prefix peering configuration.

The incorrect setting of BGP Roles and/or OTC Attributes may affect prefix propagation. Further, this document does not specify any special handling of an incorrect AS number in the OTC Attribute.

In AS migration scenarios [RFC7705], a given router may represent itself as any one of several different ASes. This should not be a problem since the egress procedures in Section 4 specify that the OTC Attribute is to be attached as part of route transmission. Therefore, a router is expected to set the OTC value equal to the ASN it is currently representing itself as.

Section 6 of [RFC7606] documents possible negative impacts of "treat-as-withdraw" behavior. Such negative impacts may include forwarding loops or blackholes. It also discusses debugging considerations related to this behavior.

6. IANA Considerations

IANA has registered a new BGP Capability (Section 3.1) in the "Capability Codes" registry's "IETF Review" range [RFC5492]. The description for the new capability is "BGP Role". IANA has assigned the value 9 [to be removed upon publication: <https://www.iana.org/assignments/capability-codes/capability-codes.xhtml>]. This document is the reference for the new capability.

The BGP Role capability includes a Value field, for which IANA is requested to create and maintain a new sub-registry called "BGP Role Value" in the Capability Codes registry. Assignments consist of a Value and a corresponding Role name. Initially, this registry is to be populated with the data contained in Table 1 found in Section 3.1. Future assignments may be made by the "IETF Review" policy as defined in [RFC8126]. The registry is as shown in Table 3.

| Value | Role name (for the local AS) | Reference |
|-------|-------------------------------|---------------|
| 0 | Provider | This document |
| 1 | RS | This document |
| 2 | RS-Client | This document |
| 3 | Customer | This document |
| 4 | Peer (i.e., Lateral Peer) | This document |
| 5-255 | To be assigned by IETF Review | |

Table 3: IANA Registry for BGP Role

IANA has registered a new OPEN Message Error subcode named the "Role Mismatch" (see Section 3.2) in the OPEN Message Error subcodes registry. IANA has assigned the value 11 [to be removed upon publication: <https://www.iana.org/assignments/bgp-parameters/bgp-parameters.xhtml#bgp-parameters-6>]. This document is the reference for the new subcode.

Due to improper use of the values 8, 9, and 10 in the OPEN Message Error subcodes registry, this document requested IANA to mark these values as "Deprecated". IANA has marked values 8-10 as "Deprecated" in the OPEN Message Error subcodes registry. This document is listed as the reference.

IANA has also registered a new path attribute named "Only to Customer (OTC)" (see Section 4) in the "BGP Path Attributes" registry. IANA has assigned code value 35 [To be removed upon publication: <http://www.iana.org/assignments/bgp-parameters/bgp-parameters.xhtml#bgp-parameters-2>]. This document is the reference for the new attribute.

7. Security Considerations

The security considerations of BGP (as specified in [RFC4271] and [RFC4272]) apply.

This document proposes a mechanism using BGP Role for the prevention and detection of route leaks that are the result of BGP policy misconfiguration. A misconfiguration of the BGP Role may affect prefix propagation. For example, if a downstream (i.e., towards a Customer) peering link were misconfigured with a Provider or Peer

Role, this will limit the number of prefixes that can be advertised in this direction. On the other hand, if an upstream provider were misconfigured (by a local AS) with the Customer Role, this may result in propagating routes that are received from other Providers or Peers. But the BGP Role negotiation and the resulting confirmation of Roles make such misconfigurations unlikely.

Setting the strict mode of operation for BGP Role negotiation as the default may result in a situation where the eBGP session will not come up after a software update. Implementations with such default behavior are strongly discouraged.

Removing the OTC Attribute or changing its value can limit the opportunity for route leak detection. Such activity can be done on purpose as part of an on-path attack. For example, an AS can remove the OTC Attribute on a received route and then leak the route to its transit provider. This kind of threat is not new in BGP and it may affect any Attribute (Note: BGPsec [RFC8205] offers protection only for the AS_PATH Attribute).

Adding an OTC Attribute when the route is advertised from Customer to Provider will limit the propagation of the route. Such a route may be considered as ineligible by the immediate Provider or its Peers or upper layer Providers. This kind of OTC Attribute addition is unlikely to happen on the Provider side because it will limit the traffic volume towards its Customer. On the Customer side, adding an OTC Attribute for traffic engineering purposes is also discouraged because it will limit route propagation in an unpredictable way.

8. References

8.1. 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>>.

- [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>>.
- [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>>.
- [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>>.
- [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>>.

8.2. Informative References

- [Gao] Gao, L. and J. Rexford, "Stable Internet routing without global coordination", IEEE/ACM Transactions on Networking, Volume 9, Issue 6, pp 689-692, DOI 10.1109/90.974523, December 2001, <<https://ieeexplore.ieee.org/document/974523>>.
- [RFC4272] Murphy, S., "BGP Security Vulnerabilities Analysis", RFC 4272, DOI 10.17487/RFC4272, January 2006, <<https://www.rfc-editor.org/info/rfc4272>>.
- [RFC7705] George, W. and S. Amante, "Autonomous System Migration Mechanisms and Their Effects on the BGP AS_PATH Attribute", RFC 7705, DOI 10.17487/RFC7705, November 2015, <<https://www.rfc-editor.org/info/rfc7705>>.
- [RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <<https://www.rfc-editor.org/info/rfc7938>>.
- [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>>.

Acknowledgments

The authors wish to thank Alvaro Retana, Bruno Decraene, Jeff Haas, John Scudder, Sue Hares, Ben Maddison, Andrei Robachevsky, Daniel Ginsburg, Ruediger Volk, Pavel Lunin, Gyan Mishra, and Ignas Bagdonas for review, comments, and suggestions during the course of this work. Thanks are also due to many IESG reviewers whose comments greatly helped improve the clarity, accuracy, and presentation in the document.

Contributors

Brian Dickson
Independent
Email: brian.peter.dickson@gmail.com

Doug Montgomery
USA National Institute of Standards and Technology
Email: dougm@nist.gov

Authors' Addresses

Alexander Azimov
Qrator Labs & Yandex
Ulitsa Iva Tolstogo 16
Moscow
119021
Russian Federation
Email: a.e.azimov@gmail.com

Eugene Bogomazov
Qrator Labs
1-y Magistralnyy tupik 5A
Moscow
123290
Russian Federation
Email: eb@qrator.net

Randy Bush
Internet Initiative Japan & Arrcus, Inc.
5147 Crystal Springs
Bainbridge Island, Washington 98110
United States of America
Email: randy@psg.com

Keyur Patel
Arrcus
2077 Gateway Place, Suite #400
San Jose, CA 95119
United States of America
Email: keyur@arrcus.com

Kotikalapudi Sriram
USA National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899
United States of America
Email: ksriram@nist.gov

IDR Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 14, 2022

A. Wang
China Telecom
H. Chen
Futurewei
K. Talaulikar
Arrcus Inc
S. Zhuang
Huawei Technologies
May 13, 2022

BGP-LS Extension for Inter-AS Topology Retrieval
draft-ietf-idr-bgpls-inter-as-topology-ext-11

Abstract

This document describes the process to build Border Gateway Protocol-Link State (BGP-LS) key parameters in inter-domain scenario, defines one new BGP-LS Network Layer Reachability Information (NLRI) type (Stub Link NLRI) and some new inter Autonomous (inter-AS) Traffic Engineering (TE) related Type-Length-Values (TLVs) for BGP-LS to let Software Definition Network (SDN) controller retrieve the network topology automatically under various inter-AS environments.

Such extension and process can enable the network operator to collect the interconnect information between different domains and then calculate the overall network topology automatically based on the information provided by BGP-LS protocol.

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 |
| 2. Conventions used in this document | 3 |
| 3. Terminology | 3 |
| 4. Inter-AS Domain Scenarios. | 3 |
| 5. Stub Link NLRI | 4 |
| 5.1. Inter-AS Native IP Scenario | 5 |
| 5.2. Inter-AS TE Scenario | 6 |
| 6. Inter-AS TE NLRI related TLVs | 6 |
| 6.1. Remote AS Number TLV | 7 |
| 6.2. IPv4 Remote ASBR ID | 7 |
| 6.3. IPv6 Remote ASBR ID | 8 |
| 7. Topology Reconstruction. | 8 |
| 8. Security Considerations | 9 |
| 9. IANA Considerations | 9 |
| 9.1. New BGP-LS NLRI type | 9 |
| 9.2. New Link Descriptors | 10 |
| 10. Acknowledgement | 10 |
| 11. References | 10 |
| 11.1. Normative References | 10 |
| 11.2. Informative References | 11 |
| Authors' Addresses | 11 |

1. Introduction

BGP-LS [RFC7752] describes the methodology that using BGP protocol to transfer the Link-State information. Such method can enable SDN controller to collect the underlay network topology automatically, but normally it can only get the information within one Interior Gateway Protocol (IGP) domain. If the operator has more than one IGP domain, and these domains interconnect with each other, there is no

mechanic within current BGP-LS to transfer the interconnect topology information.

Draft [I-D.ietf-idr-bgppls-segment-routing-epe] defines some extensions for exporting BGP peering node topology information (including its peers, interfaces and peering ASs) in a way that is exploitable in order to compute efficient BGP Peering Engineering policies and strategies. Such information can also be used to calculate the interconnection topology among different IGP domains, but it requires every border router to run BGP-LS protocol and report the information to SDN controller. Considering there will be several border routers on the network boundary, such solution restricts its deployment flexibility.

This draft analysis the situations that the SDN controller needs to get the interconnected topology information between different AS domains, defines the new Stub Link NLRI and some new TLVs within the BGP-LS protocol to transfer the key information related to them. After that, the SDN controller can then deduce the multi-domain topology automatically based on the information from BGP-LS protocol.

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] .

3. Terminology

The following terms are defined in this document:

- o IDCs: Internet Data Centers
- o MAN: Metrio-Area-Network
- o SDN: Software Definition Network

4. Inter-AS Domain Scenarios.

Figure 1 illustrates the multi-domain scenarios that this draft discusses. Normally, SDN Controller can get the topology of IGP A and IGP B individually via the BGP-LS protocol, but it can't get the topology connection information between these two IGP domains because there is generally no IGP protocol run on the connected links.

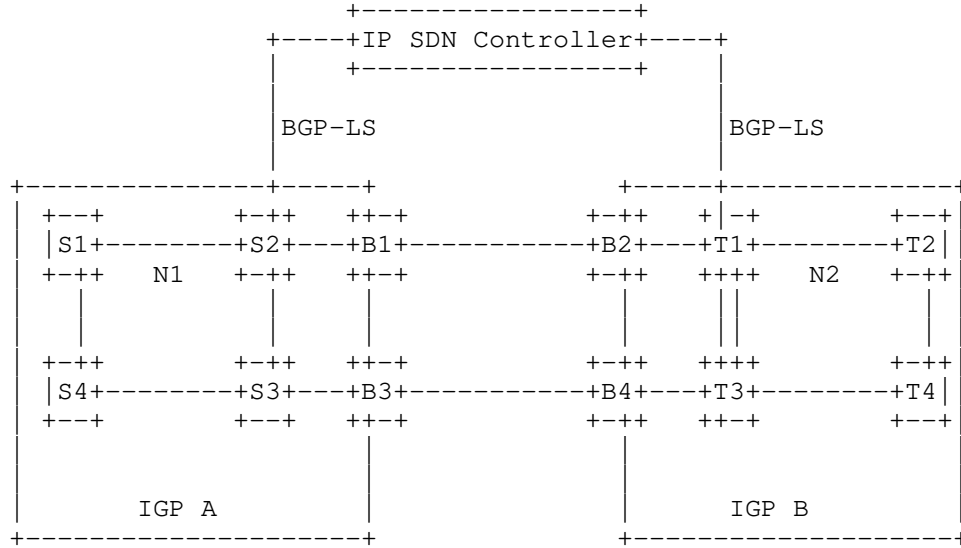


Figure 1: Inter-AS Domain Scenarios

5. Stub Link NLRI

[RFC7752] defines four NLRI types(Node NLRI, Link NLRI, IPv4 Topology Prefix NLRI, IPv6 Topology Prefix NLRI) to transfer the topology and prefix information. For inter-as link, the two ends of the link locates in different IGP domains, then it is not appropriate to transfer their information within the current defined NLRI types.

This draft defines one new NLRI type, called Stub Link NLRI, which is coded as the following format:

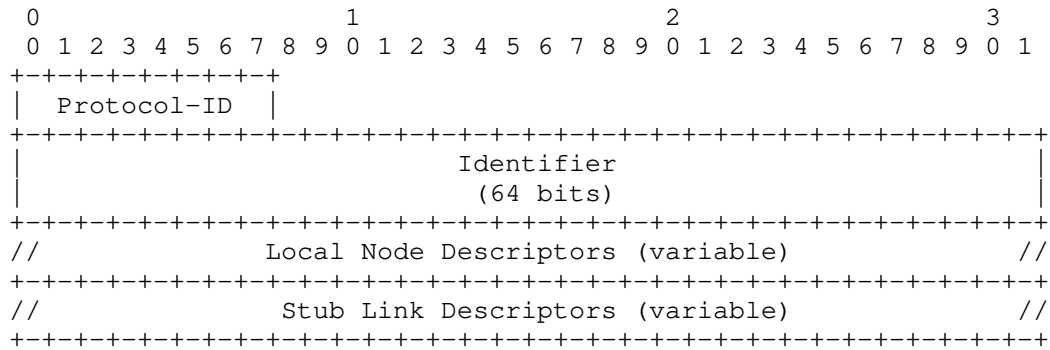


Figure 2: Stub Link NLRI Format

The "Protocol-ID" should be set to the value that indicates "Direct" protocol.

The semantics of "Local Node Descriptors" and "Stub Link Descriptors" are same as that defined in [RFC7752] for "Node Descriptors" and "Link Descriptor".

This newly defined NLRI can be used to describe the link that has only one end located within the IGP domain, as described in the following sections.

5.1. Inter-AS Native IP Scenario

Draft [RFC8735] describes the situation that operator needs some traffic engineering solution for the inter-as native IP environment. In such situation, different domain may run different IGP protocol. The operator needs to know the inter-as topology first to calculate the end to end optimal path centrally.

When IGP A or IGP B in Figure 1 runs native IS-IS/OSPF protocol, the operator can use passive feature for the inter-domain links to let the routers within the IGP domain know these links. Such stub links information can then be carried within the Stub Link NLRI reported via the BGP-LS protocol to the SDN controller.

For OSPFv2, when the interface is configured as passive, the "Linktype" field in corresponding Router LSA will be set to 3, to indicate it connects with stub network. Other routers in the IGP domain can identify such interfaces via this characteristics, and report them via the newly defined "Stub Link NLRI".

For OSPFv3 and ISIS, [I-D.wang-lsr-stub-link-attributes] describes the method to identify the stub link within the network. The router that runs BGP-LS can extract these stub links from other interfaces that participate in the IGP protocol and report them via the newly defined "Stub Link NLRI".

The "Local Node Descriptors" should describe the characteristics of ASBRs that are connected these stub links.

When such information is reported via the BGP-LS protocol, the SDN controller can construct the underlay inter-domain topology according to procedure described in Section 7

5.2. Inter-AS TE Scenario

When IGP A or IGP B in Figure 1 runs IS-IS TE/OSPF-TE protocol, [RFC5316] and [RFC5392] define IS-IS and OSPF extensions respectively to deal with the situation for inter-AS traffic engineering. Three new sub-TLVs (Remote AS Number; IPv4 Remote ASBR ID; IPv6 Remote ASBR ID) which are associated with the inter-AS TE link are defined.

These TLVs are flooded within the IGP domain automatically. They should be carried within the newly defined Stub Link NLRI within the BGP-LS protocol, as the descriptors for the inter-AS stub link.

The "Local Node Descriptors" should describe the characteristics of ASBRs that are connected these inter-AS TE links.

If the SDN controller knows these information via one of the interior router that runs BGP-LS protocol, the SDN controller can rebuild the inter-AS TE topology correctly according to the procedure described in Section 7

6. Inter-AS TE NLRI related TLVs

This draft proposes to add three new TLVs that is included within the Stub Link NLRI to transfer the information via BGP-LS, which are required to build the inter-AS TE related topology by the SDN controller.

The following Link Descriptor TLVs are added into the BGP-LS protocol :

| TLV Code Point | Description | IS-IS/OSPF TLV /Sub-TLV | Reference (RFC/Section) |
|----------------|---------------------|-------------------------|-------------------------|
| TBD | Remote AS Number | 24/21 | [RFC5316]/3.3.1 |
| TBD | IPv4 Remote ASBR ID | 25/22 | [RFC5392]/3.3.1 |
| TBD | IPv6 Remote ASBR ID | 26/24 | [RFC5316]/3.3.2 |
| | | | [RFC5392]/3.3.2 |
| | | | [RFC5316]/3.3.3 |
| | | | [RFC5392]/3.3.3 |

Figure 3: Link Descriptor TLVs

Detail encoding of these TLVs are synchronized with the corresponding parts in [RFC5316] and [RFC5392], which keeps the BGP-LS protocol agnostic to the underly protocol.

6.1. Remote AS Number TLV

A new TLV, the remote AS number TLV, is defined for inclusion in the link descriptor when advertising inter-AS TE links. The remote AS number TLV specifies the AS number of the neighboring AS to which the advertised link connects.

The remote AS number TLV is TLV type TBD (see Section 9) and is 4 octets in length. The format is as follows:

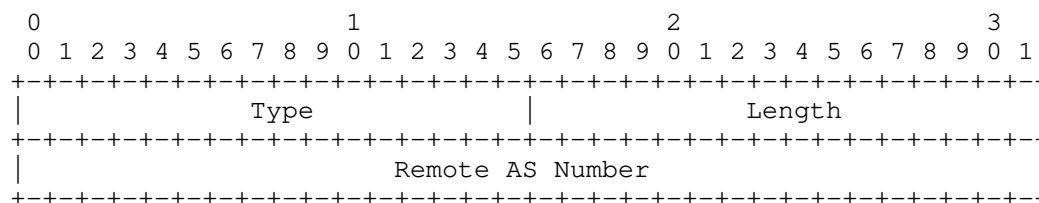


Figure 4: Remote AS Number TLV Format

The Remote AS number field has 4 octets. When only 2 octets are used for the AS number, as in current deployments, the left (high-order) 2 octets MUST be set to 0. The remote AS number TLV MUST be included when a router advertises an inter-AS TE link.

6.2. IPv4 Remote ASBR ID

A new TLV, which is referred to as the IPv4 remote ASBR ID TLV, is defined for inclusion in the link descriptor when advertising inter-AS TE links. The IPv4 remote ASBR ID TLV specifies the IPv4 identifier of the remote ASBR to which the advertised inter-AS link connects. This could be any stable and routable IPv4 address of the remote ASBR. Use of the TE Router ID as specified in the Traffic Engineering router ID TLV [RFC5316] is RECOMMENDED.

The IPv4 remote ASBR ID TLV is TLV type TBD (see Section 9) and is 4 octets in length. The format of the IPv4 remote ASBR ID sub-TLV is as follows:

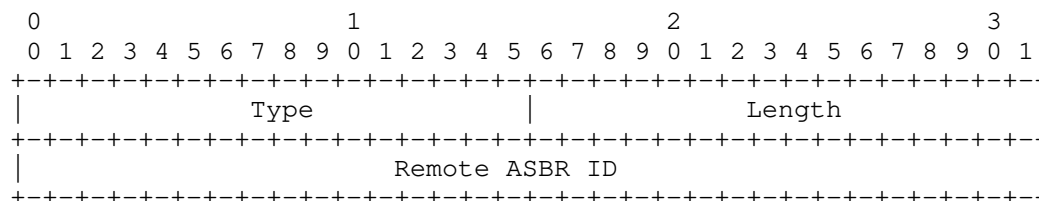


Figure 5: IPv4 Remote ASBR ID TLV Format

The IPv4 remote ASBR ID TLV MUST be included if the neighboring ASBR has an IPv4 address. If the neighboring ASBR does not have an IPv4 address (not even an IPv4 TE Router ID), the IPv6 remote ASBR ID TLV MUST be included instead. An IPv4 remote ASBR ID TLV and IPv6 remote ASBR ID TLV MAY both be present in an inter-AS TE link NLRI.

6.3. IPv6 Remote ASBR ID

A new TLV, which is referred to as the IPv6 remote ASBR ID TLV, is defined for inclusion in the link descriptor when advertising inter-AS links. The IPv6 remote ASBR ID TLV specifies the IPv6 identifier of the remote ASBR to which the advertised inter-AS link connects. This could be any stable and routable IPv6 address of the remote ASBR. Use of the TE Router ID as specified in the IPv6 Traffic Engineering router ID TLV [RFC5316] is RECOMMENDED.

The IPv6 remote ASBR ID TLV is TLV type TBD (see Section 9) and is 16 octets in length. The format of the IPv6 remote ASBR ID TLV is as follows:

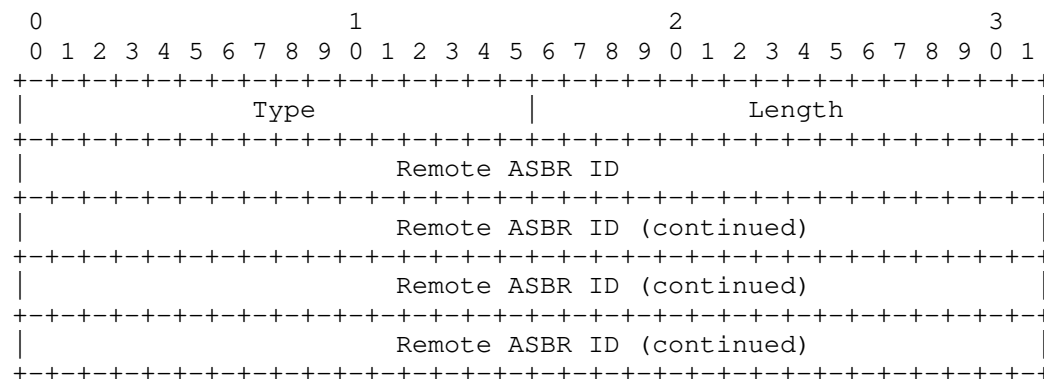


Figure 6: IPv6 Remote ASBR ID TLV Format

The IPv6 remote ASBR ID TLV MUST be included if the neighboring ASBR has an IPv6 address. If the neighboring ASBR does not have an IPv6 address, the IPv4 remote ASBR ID TLV MUST be included instead. An IPv4 remote ASBR ID TLV and IPv6 remote ASBR ID TLV MAY both be present in an inter-AS TE link NLRI.

7. Topology Reconstruction.

When SDN controller gets such information from BGP-LS protocol, it should compares the proximity of these stub links. If they are under the same network scope and in different AS, then it should find the corresponding associated router information, build the link between these two border routers.

If the prefixes reported via the "Stub Link" NLRI are under the same network scope, and in the same AS, the SDN controller can then determine there is some IGP adjacency irregular. The usage of such information is out of scope of this draft.

After iterating the above procedures for all of the stub links, the SDN controller can then retrieve the connection topology between different domains automatically.

8. Security Considerations

It is common for one operator to occupy several IGP domains that are composited by its backbone network and several MAN (Metropolitan Area Network)s/Internet Data Centers (IDCs). When they do traffic engineering which spans MAN, Backbone and IDC, they need to know the inter-as topology via the process described in this draft. Using the passive interface features or configuring the Traffic Engineering (TE) parameters on the interconnect links will not spread the topology fluctuation across each other domain.

9. IANA Considerations

This document defines:

- o A new BGP NLRI Type: Stub Link NLRI. The codepoint is from the "BGP-LS NLRI Types"
- o Three new Link Descriptors TLV: Remote AS Number TLV, IPv4 Remote ASBR ID, IPv6 Remote ASBR ID. The codepoint are from "BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs" registry.

9.1. New BGP-LS NLRI type

This document defines a new value in the registry "BGP-LS NLRI Types":

| Code Point | Description | Status |
|------------|----------------|----------------------|
| TBD | Stub Link NLRI | Allocation from IANA |

Figure 7: Stub Link NLRI Codepoint

9.2. New Link Descriptors

This document defines three new values in the registry "BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs":

| Code Point | Description | Status |
|------------|---------------------|----------------------|
| TBD | Remote AS Number | Allocation from IANA |
| TBD | IPv4 Remote ASBR ID | Allocation from IANA |
| TBD | IPv6 Remote ASBR ID | Allocation from IANA |

Figure 8: BGP-LS Link Descriptors TLV

10. Acknowledgement

The author would like to thank Acee Lindem, Jie Dong, Shaowen Ma, Jeff Tantsura and Dhruv Dhody for their valuable comments and suggestions.

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>>.
- [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>>.
- [RFC5316] Chen, M., Zhang, R., and X. Duan, "ISIS Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5316, DOI 10.17487/RFC5316, December 2008, <<https://www.rfc-editor.org/info/rfc5316>>.
- [RFC5392] Chen, M., Zhang, R., and X. Duan, "OSPF Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5392, DOI 10.17487/RFC5392, January 2009, <<https://www.rfc-editor.org/info/rfc5392>>.

- [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>>.
- [RFC8735] Wang, A., Huang, X., Kou, C., Li, Z., and P. Mi, "Scenarios and Simulation Results of PCE in a Native IP Network", RFC 8735, DOI 10.17487/RFC8735, February 2020, <<https://www.rfc-editor.org/info/rfc8735>>.

11.2. Informative References

- [I-D.ietf-idr-bgpls-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", draft-ietf-idr-bgpls-segment-routing-epe-19 (work in progress), May 2019.
- [I-D.wang-lsr-stub-link-attributes]
Wang, A., Hu, Z., Mishra, G. S., Lindem, A., and J. Sun, "Advertisement of Stub Link Attributes", draft-wang-lsr-stub-link-attributes-03 (work in progress), January 2022.

Authors' Addresses

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing, Beijing 102209
China

Email: wangaj3@chinatelecom.cn

Huaimo Chen
Futurewei
Boston, MA
USA

Email: hchen@futurewei.com

Ketan Talaulikar
Arrcus Inc
India

Email: ketant.ietf@gmail.com

Shunwan Zhuang
Huawei Technologies
Huawei Building, No.156 Beiqing Rd.
Beijing 100095
China

Email: zhuangshunwan@huawei.com

IDR Working Group
Internet-Draft
Obsoletes: 5512, 5566 (if approved)
Updates: 5640 (if approved)
Intended status: Standards Track
Expires: July 11, 2021

K. Patel
Arrcus, Inc
G. Van de Velde
Nokia
S. Sangli
J. Scudder
Juniper Networks
January 7, 2021

The BGP Tunnel Encapsulation Attribute
draft-ietf-idr-tunnel-encaps-22

Abstract

This document defines a BGP Path Attribute known as the "Tunnel Encapsulation Attribute", which can be used with BGP UPDATES of various SAFIs to provide information needed to create tunnels and their corresponding encapsulation headers. It provides encodings for a number of Tunnel Types along with procedures for choosing between alternate tunnels and routing packets into tunnels.

This document obsoletes RFC 5512, which provided an earlier definition of the Tunnel Encapsulation Attribute. RFC 5512 was never deployed in production. Since RFC 5566 relies on RFC 5512, it is likewise obsoleted. This document updates RFC 5640 by indicating that the Load-Balancing Block sub-TLV may be included in any Tunnel Encapsulation Attribute where load balancing is desired.

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 July 11, 2021.

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 | 3 |
| 1.1. Brief Summary of RFC 5512 | 4 |
| 1.2. Deficiencies in RFC 5512 | 4 |
| 1.3. Use Case for The Tunnel Encapsulation Attribute | 5 |
| 1.4. Brief Summary of Changes from RFC 5512 | 6 |
| 1.5. Update to RFC 5640 | 7 |
| 1.6. Effects of Obsoleting RFC 5566 | 7 |
| 2. The Tunnel Encapsulation Attribute | 8 |
| 3. Tunnel Encapsulation Attribute Sub-TLVs | 9 |
| 3.1. The Tunnel Egress Endpoint Sub-TLV (type code 6) | 9 |
| 3.1.1. Validating the Address Subfield | 11 |
| 3.2. Encapsulation Sub-TLVs for Particular Tunnel Types (type code 1) | 12 |
| 3.2.1. VXLAN (tunnel type 8) | 12 |
| 3.2.2. NVGRE (tunnel type 9) | 14 |
| 3.2.3. L2TPv3 (tunnel type 1) | 16 |
| 3.2.4. GRE (tunnel type 2) | 16 |
| 3.2.5. MPLS-in-GRE (tunnel type 11) | 17 |
| 3.3. Outer Encapsulation Sub-TLVs | 17 |
| 3.3.1. DS Field (type code 7) | 18 |
| 3.3.2. UDP Destination Port (type code 8) | 18 |
| 3.4. Sub-TLVs for Aiding Tunnel Selection | 19 |
| 3.4.1. Protocol Type Sub-TLV (type code 2) | 19 |
| 3.4.2. Color Sub-TLV (type code 4) | 20 |
| 3.5. Embedded Label Handling Sub-TLV (type code 9) | 20 |
| 3.6. MPLS Label Stack Sub-TLV (type code 10) | 21 |
| 3.7. Prefix-SID Sub-TLV (type code 11) | 23 |
| 4. Extended Communities Related to the Tunnel Encapsulation Attribute | 24 |
| 4.1. Encapsulation Extended Community | 24 |
| 4.2. Router's MAC Extended Community | 25 |

| | |
|---|----|
| 4.3. Color Extended Community | 26 |
| 5. Special Considerations for IP-in-IP Tunnels | 26 |
| 6. Semantics and Usage of the Tunnel Encapsulation attribute . . | 26 |
| 7. Routing Considerations | 29 |
| 7.1. Impact on the BGP Decision Process | 29 |
| 7.2. Looping, Mutual Recursion, Etc. | 29 |
| 8. Recursive Next Hop Resolution | 30 |
| 9. Use of Virtual Network Identifiers and Embedded Labels when Imposing a Tunnel Encapsulation | 31 |
| 9.1. Tunnel Types without a Virtual Network Identifier Field . | 31 |
| 9.2. Tunnel Types with a Virtual Network Identifier Field . . | 31 |
| 9.2.1. Unlabeled Address Families | 32 |
| 9.2.2. Labeled Address Families | 32 |
| 10. Applicability Restrictions | 33 |
| 11. Scoping | 34 |
| 12. Operational Considerations | 35 |
| 13. Validation and Error Handling | 35 |
| 14. IANA Considerations | 36 |
| 14.1. Obsoleting RFC 5512 | 36 |
| 14.2. Obsoleting Code Points Assigned by RFCs 5566 | 37 |
| 14.3. BGP Tunnel Encapsulation Parameters Grouping | 37 |
| 14.4. BGP Tunnel Encapsulation Attribute Tunnel Types | 37 |
| 14.5. Subsequent Address Family Identifiers | 37 |
| 14.6. BGP Tunnel Encapsulation Attribute Sub-TLVs | 37 |
| 14.7. Flags Field of VXLAN Encapsulation sub-TLV | 38 |
| 14.8. Flags Field of NVGRE Encapsulation sub-TLV | 39 |
| 14.9. Embedded Label Handling sub-TLV | 39 |
| 14.10. Color Extended Community Flags | 39 |
| 15. Security Considerations | 40 |
| 16. Acknowledgments | 41 |
| 17. Contributor Addresses | 41 |
| 18. References | 42 |
| 18.1. Normative References | 42 |
| 18.2. Informative References | 44 |
| Appendix A. Impact on RFC 8365 | 46 |
| Authors' Addresses | 46 |

1. Introduction

This document obsoletes RFC 5512. The deficiencies of RFC 5512, and a summary of the changes made, are discussed in Sections 1.1-1.3. The material from RFC 5512 that is retained has been incorporated into this document. Since [RFC5566] relies on RFC 5512, it is likewise obsoleted.

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.

1.1. Brief Summary of RFC 5512

[RFC5512] defines a BGP Path Attribute known as the Tunnel Encapsulation attribute. This attribute consists of one or more TLVs. Each TLV identifies a particular type of tunnel. Each TLV also contains one or more sub-TLVs. Some of the sub-TLVs, for example, the "Encapsulation sub-TLV", contain information that may be used to form the encapsulation header for the specified Tunnel Type. Other sub-TLVs, for example, the "color sub-TLV" and the "protocol sub-TLV", contain information that aids in determining whether particular packets should be sent through the tunnel that the TLV identifies.

[RFC5512] only allows the Tunnel Encapsulation attribute to be attached to BGP UPDATE messages of the Encapsulation Address Family. These UPDATE messages have an AFI (Address Family Identifier) of 1 or 2, and a SAFI of 7. In an UPDATE of the Encapsulation SAFI, the NLRI (Network Layer Reachability Information) is an address of the BGP speaker originating the UPDATE. Consider the following scenario:

- o BGP speaker R1 has received and selected UPDATE U for local use;
- o UPDATE U's SAFI is the Encapsulation SAFI;
- o UPDATE U has the address R2 as its NLRI;
- o UPDATE U has a Tunnel Encapsulation attribute.
- o R1 has a packet, P, to transmit to destination D;
- o R1's best route to D is a BGP route that has R2 as its next hop;

In this scenario, when R1 transmits packet P, it should transmit it to R2 through one of the tunnels specified in U's Tunnel Encapsulation attribute. The IP address of the tunnel egress endpoint of each such tunnel is R2. Packet P is known as the tunnel's "payload".

1.2. Deficiencies in RFC 5512

While the ability to specify tunnel information in a BGP UPDATE is useful, the procedures of [RFC5512] have certain limitations:

- o The requirement to use the "Encapsulation SAFI" presents an unfortunate operational cost, as each BGP session that may need to

carry tunnel encapsulation information needs to be reconfigured to support the Encapsulation SAFI. The Encapsulation SAFI has never been used, and this requirement has served only to discourage the use of the Tunnel Encapsulation attribute.

- o There is no way to use the Tunnel Encapsulation attribute to specify the tunnel egress endpoint address of a given tunnel; [RFC5512] assumes that the tunnel egress endpoint of each tunnel is specified as the NLRI of an UPDATE of the Encapsulation SAFI.
- o If the respective best routes to two different address prefixes have the same next hop, [RFC5512] does not provide a straightforward method to associate each prefix with a different tunnel.
- o If a particular Tunnel Type requires an outer IP or UDP encapsulation, there is no way to signal the values of any of the fields of the outer encapsulation.
- o In [RFC5512]'s specification of the sub-TLVs, each sub-TLV has one-octet length field. In some cases, where a sub-TLV may require more than 255 octets for its encoding, a two-octet length field may be needed.

1.3. Use Case for The Tunnel Encapsulation Attribute

Consider the case of a router R1 forwarding an IP packet P. Let D be P's IP destination address. R1 must look up D in its forwarding table. Suppose that the "best match" route for D is route Q, where Q is a BGP-distributed route whose "BGP next hop" is router R2. And suppose further that the routers along the path from R1 to R2 have entries for R2 in their forwarding tables, but do NOT have entries for D in their forwarding tables. For example, the path from R1 to R2 may be part of a "BGP-free core", where there are no BGP-distributed routes at all in the core. Or, as in [RFC5565], D may be an IPv4 address while the intermediate routers along the path from R1 to R2 may support only IPv6.

In cases such as this, in order for R1 to properly forward packet P, it must encapsulate P and send P "through a tunnel" to R2. For example, R1 may encapsulate P using GRE, L2TPv3, IP in IP, etc., where the destination IP address of the encapsulation header is the address of R2.

In order for R1 to encapsulate P for transport to R2, R1 must know what encapsulation protocol to use for transporting different sorts of packets to R2. R1 must also know how to fill in the various fields of the encapsulation header. With certain encapsulation

types, this knowledge may be acquired by default or through manual configuration. Other encapsulation protocols have fields such as session id, key, or cookie that must be filled in. It would not be desirable to require every BGP speaker to be manually configured with the encapsulation information for every one of its BGP next hops.

This document specifies a way in which BGP itself can be used by a given BGP speaker to tell other BGP speakers, "if you need to encapsulate packets to be sent to me, here's the information you need to properly form the encapsulation header". A BGP speaker signals this information to other BGP speakers by using a new BGP attribute type value, the BGP Tunnel Encapsulation Attribute. This attribute specifies the encapsulation protocols that may be used as well as whatever additional information (if any) is needed in order to properly use those protocols. Other attributes, for example, communities or extended communities, may also be included.

1.4. Brief Summary of Changes from RFC 5512

This document addresses the deficiencies identified in Section 1.2 by:

- o Deprecating the Encapsulation SAFI.
- o Defining a new "Tunnel Egress Endpoint sub-TLV" (Section 3.1) that can be included in any of the TLVs contained in the Tunnel Encapsulation attribute. This sub-TLV can be used to specify the remote endpoint address of a particular tunnel.
- o Allowing the Tunnel Encapsulation attribute to be carried by BGP UPDATES of additional AFI/SAFIs. Appropriate semantics are provided for this way of using the attribute.
- o Defining a number of new sub-TLVs that provide additional information that is useful when forming the encapsulation header used to send a packet through a particular tunnel.
- o Defining the sub-TLV type field so that a sub-TLV whose type is in the range from 0 to 127 inclusive has a one-octet length field, but a sub-TLV whose type is in the range from 128 to 255 inclusive has a two-octet length field.

One of the sub-TLVs defined in [RFC5512] is the "Encapsulation sub-TLV". For a given tunnel, the Encapsulation sub-TLV specifies some of the information needed to construct the encapsulation header used when sending packets through that tunnel. This document defines Encapsulation sub-TLVs for a number of tunnel types not discussed in [RFC5512]: VXLAN (Virtual Extensible Local Area Network, [RFC7348]),

NVGRE (Network Virtualization Using Generic Routing Encapsulation [RFC7637]), and MPLS-in-GRE (MPLS in Generic Routing Encapsulation [RFC4023]). MPLS-in-UDP [RFC7510] is also supported, but an Encapsulation sub-TLV for it is not needed since there are no additional parameters to be signaled.

Some of the encapsulations mentioned in the previous paragraph need to be further encapsulated inside UDP and/or IP. [RFC5512] provides no way to specify that certain information is to appear in these outer IP and/or UDP encapsulations. This document provides a framework for including such information in the TLVs of the Tunnel Encapsulation attribute.

When the Tunnel Encapsulation attribute is attached to a BGP UPDATE whose AFI/SAFI identifies one of the labeled address families, it is not always obvious whether the label embedded in the NLRI is to appear somewhere in the tunnel encapsulation header (and if so, where), or whether it is to appear in the payload, or whether it can be omitted altogether. This is especially true if the tunnel encapsulation header itself contains a "virtual network identifier". This document provides a mechanism that allows one to signal (by using sub-TLVs of the Tunnel Encapsulation attribute) how one wants to use the embedded label when the tunnel encapsulation has its own virtual network identifier field.

[RFC5512] defines a Tunnel Encapsulation Extended Community that can be used instead of the Tunnel Encapsulation attribute under certain circumstances. This document describes (Section 4.1) how the Tunnel Encapsulation Extended Community can be used in a backwards-compatible fashion. It is possible to combine Tunnel Encapsulation Extended Communities and Tunnel Encapsulation attributes in the same BGP UPDATE in this manner.

1.5. Update to RFC 5640

This document updates [RFC5640] by indicating that the Load-Balancing Block sub-TLV MAY be included in any Tunnel Encapsulation Attribute where loadbalancing is desired.

1.6. Effects of Obsoleting RFC 5566

This specification obsoletes RFC 5566. This has the effect of, in turn, obsoleting a number of code points defined in that document. From the "BGP Tunnel Encapsulation Attribute Tunnel Types" registry, "Transmit tunnel endpoint" (type code 3), "IPsec in Tunnel-mode" (type code 4), "IP in IP tunnel with IPsec Transport Mode" (type code 5), and "MPLS-in-IP tunnel with IPsec Transport Mode" (type code 6) are obsoleted. From the "BGP Tunnel Encapsulation Attribute Sub-

TLVs" registry, "IPsec Tunnel Authenticator" (type code 3) is obsoleted. See Section 14.2.

2. The Tunnel Encapsulation Attribute

The Tunnel Encapsulation attribute is an optional transitive BGP Path attribute. IANA has assigned the value 23 as the type code of the attribute. The attribute is composed of a set of Type-Length-Value (TLV) encodings. Each TLV contains information corresponding to a particular Tunnel Type. A Tunnel Encapsulation TLV, also known as Tunnel TLV, is structured as shown in Figure 1:

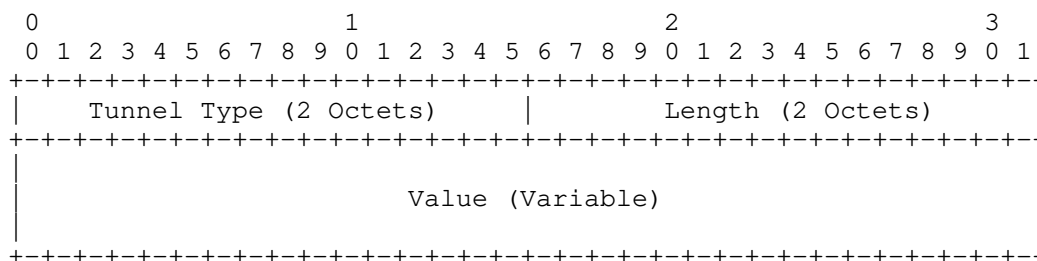


Figure 1: Tunnel Encapsulation TLV Value Field

- o Tunnel Type (2 octets): identifies a type of tunnel. The field contains values from the IANA Registry "BGP Tunnel Encapsulation Attribute Tunnel Types". See Section 3.4.1 for discussion of special treatment of tunnel types with names of the form "X-in-Y".
- o Length (2 octets): the total number of octets of the Value field.
- o Value (variable): comprised of multiple sub-TLVs.

Each sub-TLV consists of three fields: a 1-octet type, a 1-octet or 2-octet length field (depending on the type), and zero or more octets of value. A sub-TLV is structured as shown in Figure 2:

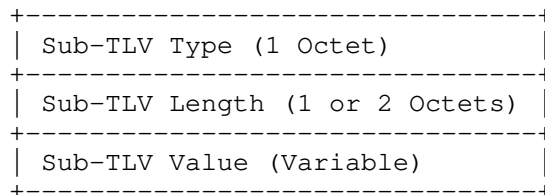


Figure 2: Encapsulation Sub-TLV Value Field

- o Sub-TLV Type (1 octet): each sub-TLV type defines a certain property about the Tunnel TLV that contains this sub-TLV. The field contains values from the IANA Registry "BGP Tunnel Encapsulation Attribute Sub-TLVs".
- o Sub-TLV Length (1 or 2 octets): the total number of octets of the sub-TLV Value field. The Sub-TLV Length field contains 1 octet if the Sub-TLV Type field contains a value in the range from 0-127. The Sub-TLV Length field contains two octets if the Sub-TLV Type field contains a value in the range from 128-255.
- o Sub-TLV Value (variable): encodings of the Value field depend on the sub-TLV type as enumerated above. The following sub-sections define the encoding in detail.

3. Tunnel Encapsulation Attribute Sub-TLVs

This section specifies a number of sub-TLVs. These sub-TLVs can be included in a TLV of the Tunnel Encapsulation attribute.

3.1. The Tunnel Egress Endpoint Sub-TLV (type code 6)

The Tunnel Egress Endpoint sub-TLV specifies the address of the egress endpoint of the tunnel, that is, the address of the router that will decapsulate the payload. Its Value field contains three subfields:

1. a reserved subfield
2. a two-octet Address Family subfield
3. an Address subfield, whose length depends upon the Address Family.

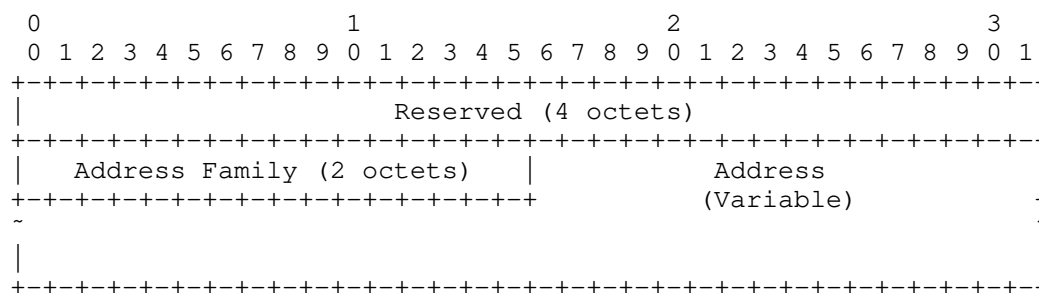


Figure 3: Tunnel Egress Endpoint Sub-TLV Value Field

The Reserved subfield SHOULD be originated as zero. It MUST be disregarded on receipt, and it MUST be propagated unchanged.

The Address Family subfield contains a value from IANA's "Address Family Numbers" registry. This document assumes that the Address Family is either IPv4 or IPv6; use of other address families is outside the scope of this document.

If the Address Family subfield contains the value for IPv4, the Address subfield MUST contain an IPv4 address (a /32 IPv4 prefix).

If the Address Family subfield contains the value for IPv6, the Address subfield MUST contain an IPv6 address (a /128 IPv6 prefix).

In a given BGP UPDATE, the address family (IPv4 or IPv6) of a Tunnel Egress Endpoint sub-TLV is independent of the address family of the UPDATE itself. For example, an UPDATE whose NLRI is an IPv4 address may have a Tunnel Encapsulation attribute containing Tunnel Egress Endpoint sub-TLVs that contain IPv6 addresses. Also, different tunnels represented in the Tunnel Encapsulation attribute may have tunnel egress endpoints of different address families.

There is one special case: the Tunnel Egress Endpoint sub-TLV MAY have a Value field whose Address Family subfield contains 0. This means that the tunnel's egress endpoint is the address of the next hop. If the Address Family subfield contains 0, the Address subfield is omitted. In this case, the Length field of Tunnel Egress Endpoint sub-TLV MUST contain the value 6 (0x06).

When the Tunnel Encapsulation attribute is carried in an UPDATE message of one of the AFI/SAFIs specified in this document (see the second paragraph of Section 6), each TLV MUST have one, and one only, Tunnel Egress Endpoint sub-TLV. If a TLV does not have a Tunnel Egress Endpoint sub-TLV, that TLV should be treated as if it had a malformed Tunnel Egress Endpoint sub-TLV (see below).

In the context of this specification, if the Address Family subfield has any value other than IPv4, IPv6, or the special value 0, the Tunnel Egress Endpoint sub-TLV is considered "unrecognized" (see Section 13). If any of the following conditions hold, the Tunnel Egress Endpoint sub-TLV is considered to be "malformed":

- o The length of the sub-TLV's Value field is other than 6 added to the defined length for the address family given in its Address Family subfield. Therefore, for address family behaviors defined in this document, the permitted values are:
 - * 10, if the Address Family subfield contains the value for IPv4.

- * 22, if the Address Family subfield contains the value for IPv6.
- * 6, if the Address Family subfield contains the value zero.
- o The IP address in the sub-TLV's Address subfield lies within a block listed in the relevant Special-Purpose IP Address Registry [RFC6890] with either a "destination" attribute value or a "forwardable" attribute value of "false". (Such routes are sometimes colloquially known as "Martians".) This restriction MAY be relaxed by explicit configuration.
- o It can be determined that the IP address in the sub-TLV's Address subfield does not belong to the Autonomous System (AS) that originated the route that contains the attribute. Section 3.1.1 describes an optional procedure to make this determination.

Error Handling is specified in Section 13.

If the Tunnel Egress Endpoint sub-TLV contains an IPv4 or IPv6 address that is valid but not reachable, the sub-TLV is not considered to be malformed.

3.1.1. Validating the Address Subfield

This section provides a procedure that MAY be applied to validate that the IP address in the sub-TLV's Address subfield belongs to the AS that originated the route that contains the attribute. (The notion of "belonging to" an AS is expanded on below.) Doing this is thought to increase confidence that when traffic is sent to the IP address depicted in the Address subfield, it will go to the same AS as it would go to if the Tunnel Encapsulation Attribute were not present, although of course it cannot guarantee it. See Section 15 for discussion of the limitations of this procedure. The principal applicability of this procedure is in deployments that are not strictly scoped. In deployments with strict scope, and especially those scoped to a single AS, these procedures may not add substantial benefit beyond those discussed in Section 11.

The Route Origin ASN (Autonomous System Number) of a BGP route that includes a Tunnel Encapsulation Attribute can be determined by inspection of the AS_PATH attribute, according to the procedure specified in [RFC6811] Section 2. Call this value Route_AS.

In order to determine the Route Origin ASN of the address depicted in the Address subfield of the Tunnel Egress Endpoint sub-TLV, it is necessary to consider the forwarding route, that is, the route that will be used to forward traffic toward that address. This route is determined by a recursive route lookup operation for that address, as

discussed in [RFC4271] Section 5.1.3. The relevant AS Path to consider is the last one encountered while performing the recursive lookup; the procedures of RFC6811 Section 2 are applied to that AS Path to determine the Route Origin ASN. If no AS Path is encountered at all, for example if that route's source is a protocol other than BGP, the Route Origin ASN is the BGP speaker's own AS number. Call this value Egress_AS.

If Route_AS does not equal Egress_AS, then the Tunnel Egress Endpoint sub-TLV is considered not to be valid. In some cases a network operator who controls a set of Autonomous Systems might wish to allow a Tunnel Egress Endpoint to reside in an AS other than Route_AS; configuration MAY allow for such a case, in which case the check becomes, if Egress_AS is not within the configured set of permitted AS numbers, then the Tunnel Egress Endpoint sub-TLV is considered to be "malformed".

Note that if the forwarding route changes, this procedure MUST be reapplied. As a result, a sub-TLV that was formerly considered valid might become not valid, or vice-versa.

3.2. Encapsulation Sub-TLVs for Particular Tunnel Types (type code 1)

This section defines Encapsulation sub-TLVs for the following tunnel types: VXLAN ([RFC7348]), NVGRE ([RFC7637]), MPLS-in-GRE ([RFC4023]), L2TPv3 ([RFC3931]), and GRE ([RFC2784]).

Rules for forming the encapsulation based on the information in a given TLV are given in Section 6 and Section 9.

Recall that the tunnel type itself is identified by the Tunnel Type field in the attribute header (Section 2); the Encapsulation sub-TLV's structure is inferred from this. Regardless of the Tunnel Type, the sub-TLV type of the Encapsulation sub-TLV is 1. There are also tunnel types for which it is not necessary to define an Encapsulation sub-TLV, because there are no fields in the encapsulation header whose values need to be signaled from the tunnel egress endpoint.

3.2.1. VXLAN (tunnel type 8)

This document defines an Encapsulation sub-TLV for VXLAN [RFC7348] tunnels. When the Tunnel Type is VXLAN, the length of the sub-TLV is 12 octets. The following is the structure of the Value field in the Encapsulation sub-TLV:

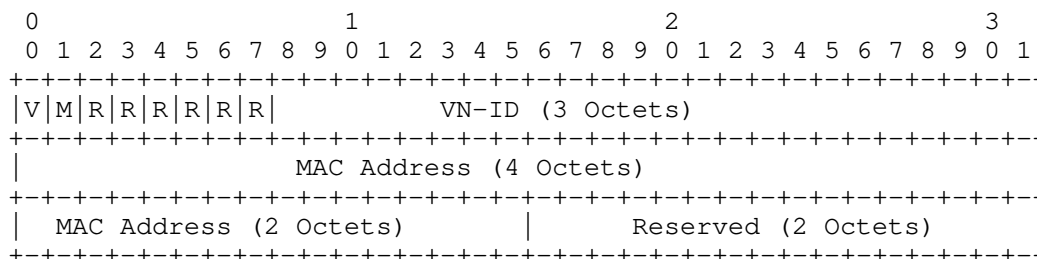


Figure 4: VXLAN Encapsulation Sub-TLV Value Field

V: This bit is set to 1 to indicate that a VN-ID (Virtual Network Identifier) is present in the Encapsulation sub-TLV. If set to 0, the VN-ID field is disregarded. Please see Section 9.

M: This bit is set to 1 to indicate that a MAC Address is present in the Encapsulation sub-TLV. If set to 0, the MAC Address field is disregarded.

R: The remaining bits in the 8-bit flags field are reserved for further use. They MUST always be set to 0 by the originator of the sub-TLV. Intermediate routers MUST propagate them without modification. Any receiving routers MUST ignore these bits upon receipt.

VN-ID: If the V bit is set, the VN-ID field contains a 3 octet VN-ID value. If the V bit is not set, the VN-ID field MUST be set to zero on transmission and disregarded on receipt.

MAC Address: If the M bit is set, this field contains a 6 octet Ethernet MAC address. If the M bit is not set, this field MUST be set to all zeroes on transmission and disregarded on receipt.

Reserved: MUST be set to zero on transmission and disregarded on receipt.

When forming the VXLAN encapsulation header:

- o The values of the V, M, and R bits are NOT copied into the flags field of the VXLAN header. The flags field of the VXLAN header is set as per [RFC7348].
- o If the M bit is set, the MAC Address is copied into the Inner Destination MAC Address field of the Inner Ethernet Header (see section 5 of [RFC7348]).

If the M bit is not set, and the payload being sent through the VXLAN tunnel is an Ethernet frame, the Destination MAC Address field of the Inner Ethernet Header is just the Destination MAC Address field of the payload's Ethernet header.

If the M bit is not set, and the payload being sent through the VXLAN tunnel is an IP or MPLS packet, the Inner Destination MAC Address field is set to a configured value; if there is no configured value, the VXLAN tunnel cannot be used.

- o If the V bit is not set, and the BGP UPDATE message has AFI/SAFI other than Ethernet VPNs (SAFI 70, "BGP EVPNs") then the VXLAN tunnel cannot be used.
- o Section 9 describes how the VNI field of the VXLAN encapsulation header is set.

Note that in order to send an IP packet or an MPLS packet through a VXLAN tunnel, the packet must first be encapsulated in an Ethernet header, which becomes the "inner Ethernet header" described in [RFC7348]. The VXLAN Encapsulation sub-TLV may contain information (for example, the MAC address) that is used to form this Ethernet header.

3.2.2. NVGRE (tunnel type 9)

This document defines an Encapsulation sub-TLV for NVGRE [RFC7637] tunnels. When the Tunnel Type is NVGRE, the length of the sub-TLV is 12 octets. The following is the structure of the Value field in the Encapsulation sub-TLV:

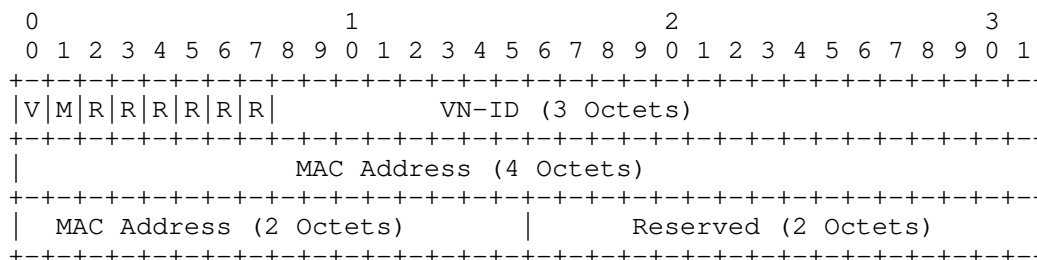


Figure 5: NVGRE Encapsulation Sub-TLV Value Field

V: This bit is set to 1 to indicate that a VN-ID is present in the Encapsulation sub-TLV. If set to 0, the VN-ID field is disregarded. Please see Section 9.

M: This bit is set to 1 to indicate that a MAC Address is present in the Encapsulation sub-TLV. If set to 0, the MAC Address field is disregarded.

R: The remaining bits in the 8-bit flags field are reserved for further use. They MUST always be set to 0 by the originator of the sub-TLV. Intermediate routers MUST propagate them without modification. Any receiving routers MUST ignore these bits upon receipt.

VN-ID: If the V bit is set, the VN-ID field contains a 3 octet VN-ID value, used to set the NVGRE VSID (see Section 9). If the V bit is not set, the VN-ID field MUST be set to zero on transmission and disregarded on receipt.

MAC Address: If the M bit is set, this field contains a 6 octet Ethernet MAC address. If the M bit is not set, this field MUST be set to all zeroes on transmission and disregarded on receipt.

Reserved: MUST be set to zero on transmission and disregarded on receipt.

When forming the NVGRE encapsulation header:

- o The values of the V, M, and R bits are NOT copied into the flags field of the NVGRE header. The flags field of the NVGRE header is set as per [RFC7637].
- o If the M bit is set, the MAC Address is copied into the Inner Destination MAC Address field of the Inner Ethernet Header (see section 3.2 of [RFC7637]).

If the M bit is not set, and the payload being sent through the NVGRE tunnel is an Ethernet frame, the Destination MAC Address field of the Inner Ethernet Header is just the Destination MAC Address field of the payload's Ethernet header.

If the M bit is not set, and the payload being sent through the NVGRE tunnel is an IP or MPLS packet, the Inner Destination MAC Address field is set to a configured value; if there is no configured value, the NVGRE tunnel cannot be used.

- o If the V bit is not set, and the BGP UPDATE message has AFI/SAFI other than Ethernet VPNs (EVPN) then the NVGRE tunnel cannot be used.
- o Section 9 describes how the VSID (Virtual Subnet Identifier) field of the NVGRE encapsulation header is set.

3.2.3. L2TPv3 (tunnel type 1)

When the Tunnel Type of the TLV is L2TPv3 over IP [RFC3931], the length of the sub-TLV is between 4 and 12 octets, depending on the length of the cookie. The following is the structure of the Value field of the Encapsulation sub-TLV:

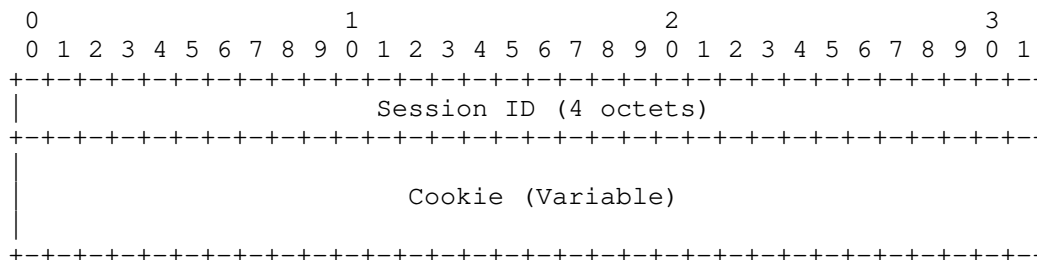


Figure 6: L2TPv3 Encapsulation Sub-TLV Value Field

Session ID: a non-zero 4-octet value locally assigned by the advertising router that serves as a lookup key for the incoming packet's context.

Cookie: an optional, variable length (encoded in octets -- 0 to 8 octets) value used by L2TPv3 to check the association of a received data message with the session identified by the Session ID. Generation and usage of the cookie value is as specified in [RFC3931].

The length of the cookie is not encoded explicitly, but can be calculated as (sub-TLV length - 4).

3.2.4. GRE (tunnel type 2)

When the Tunnel Type of the TLV is GRE [RFC2784], the length of the sub-TLV is 4 octets. The following is the structure of the Value field of the Encapsulation sub-TLV:

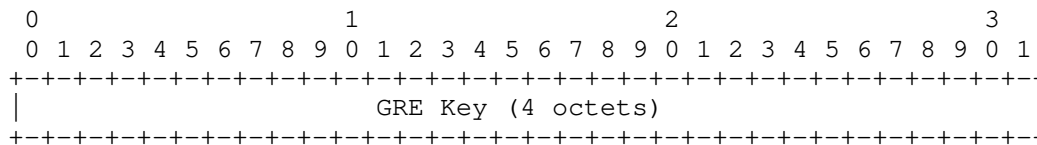


Figure 7: GRE Encapsulation Sub-TLV

GRE Key: 4-octet field [RFC2890] that is generated by the advertising router. Note that the key is optional. Unless a key

value is being advertised, the GRE Encapsulation sub-TLV MUST NOT be present.

3.2.5. MPLS-in-GRE (tunnel type 11)

When the Tunnel Type is MPLS-in-GRE [RFC4023], the length of the sub-TLV is 4 octets. The following is the structure of the Value field of the Encapsulation sub-TLV:

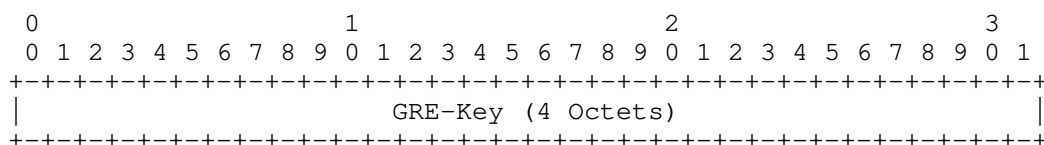


Figure 8: MPLS-in-GRE Encapsulation Sub-TLV Value Field

GRE-Key: 4-octet field [RFC2890] that is generated by the advertising router. Note that the key is optional. Unless a key value is being advertised, the MPLS-in-GRE Encapsulation sub-TLV MUST NOT be present.

Note that the GRE Tunnel Type defined in Section 3.2.4 can be used instead of the MPLS-in-GRE Tunnel Type when it is necessary to encapsulate MPLS in GRE. Including a TLV of the MPLS-in-GRE tunnel type is equivalent to including a TLV of the GRE Tunnel Type that also includes a Protocol Type sub-TLV (Section 3.4.1) specifying MPLS as the protocol to be encapsulated.

Although the MPLS-in-GRE tunnel type is just a special case of the GRE tunnel type and thus is not strictly necessary, it is included for reasons of backwards compatibility with, for example, implementations of [RFC8365].

3.3. Outer Encapsulation Sub-TLVs

The Encapsulation sub-TLV for a particular Tunnel Type allows one to specify the values that are to be placed in certain fields of the encapsulation header for that Tunnel Type. However, some tunnel types require an outer IP encapsulation, and some also require an outer UDP encapsulation. The Encapsulation sub-TLV for a given Tunnel Type does not usually provide a way to specify values for fields of the outer IP and/or UDP encapsulations. If it is necessary to specify values for fields of the outer encapsulation, additional sub-TLVs must be used. This document defines two such sub-TLVs.

If an outer Encapsulation sub-TLV occurs in a TLV for a Tunnel Type that does not use the corresponding outer encapsulation, the sub-TLV MUST be treated as if it were an unrecognized type of sub-TLV.

3.3.1. DS Field (type code 7)

Most of the tunnel types that can be specified in the Tunnel Encapsulation attribute require an outer IP encapsulation. The Differentiated Services (DS) Field sub-TLV can be carried in the TLV of any such Tunnel Type. It specifies the setting of the one-octet Differentiated Services field in the outer IPv4 or IPv6 encapsulation (see [RFC2474]). Any one-octet value can be transported; the semantics of the DSCP field is beyond the scope of this document. The Value field is always a single octet.

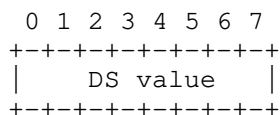


Figure 9: DS Field Sub-TLV Value Field

Because the interpretation of the DSCP field at the recipient may be different from its interpretation at the originator, an implementation MAY provide a facility to use policy to filter or modify the DS Field.

3.3.2. UDP Destination Port (type code 8)

Some of the tunnel types that can be specified in the Tunnel Encapsulation attribute require an outer UDP encapsulation. Generally there is a standard UDP Destination Port value for a particular Tunnel Type. However, sometimes it is useful to be able to use a non-standard UDP destination port. If a particular tunnel type requires an outer UDP encapsulation, and it is desired to use a UDP destination port other than the standard one, the port to be used can be specified by including a UDP Destination Port sub-TLV. The Value field of this sub-TLV is always a two-octet field, containing the port value. Any two-octet value other than zero can be transported. If the reserved value zero is received, the sub-TLV MUST be treated as malformed according to the rules of Section 13.

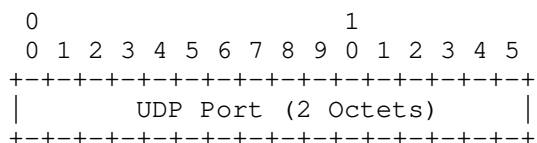


Figure 10: UDP Destination Port Sub-TLV Value Field

3.4. Sub-TLVs for Aiding Tunnel Selection

3.4.1. Protocol Type Sub-TLV (type code 2)

The Protocol Type sub-TLV MAY be included in a given TLV to indicate the type of the payload packets that are allowed to be encapsulated with the tunnel parameters that are being signaled in the TLV. Packets with other payload types MUST NOT be encapsulated in the relevant tunnel. The Value field of the sub-TLV contains a 2-octet value from IANA's "ETHER TYPES" registry [Ethertypes]. If the reserved value 0xFFFF is received, the sub-TLV MUST be treated as malformed according to the rules of Section 13.

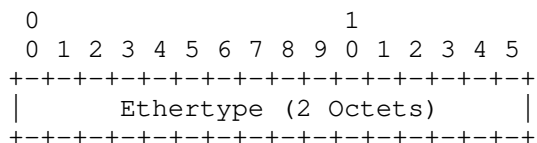


Figure 11: Protocol Type Sub-TLV Value Field

For example, if there are three L2TPv3 sessions, one carrying IPv4 packets, one carrying IPv6 packets, and one carrying MPLS packets, the egress router will include three TLVs of L2TPv3 encapsulation type, each specifying a different Session ID and a different payload type. The Protocol Type sub-TLV for these will be IPv4 (protocol type = 0x0800), IPv6 (protocol type = 0x86dd), and MPLS (protocol type = 0x8847), respectively. This informs the ingress routers of the appropriate encapsulation information to use with each of the given protocol types. Insertion of the specified Session ID at the ingress routers allows the egress to process the incoming packets correctly, according to their protocol type.

Note that for tunnel types whose names are of the form "X-in-Y", for example, "MPLS-in-GRE", only packets of the specified payload type "X" are to be carried through the tunnel of type "Y". This is the equivalent of specifying a Tunnel Type "Y" and including in its TLV a Protocol Type sub-TLV (see Section 3.4.1) specifying protocol "X". If the Tunnel Type is "X-in-Y", it is unnecessary, though harmless, to explicitly include a Protocol Type sub-TLV specifying "X". Also,

for "X-in-Y" type tunnels, a Protocol Type sub-TLV specifying anything other than "X" MUST be ignored; this is discussed further in Section 13.

3.4.2. Color Sub-TLV (type code 4)

The Color sub-TLV MAY be used as a way to "color" the corresponding Tunnel TLV. The Value field of the sub-TLV is eight octets long, and consists of a Color Extended Community, as defined in Section 4.3. For the use of this sub-TLV and Extended Community, please see Section 8.

The format of the Value field is depicted in Figure 15.

If the Length field of a Color sub-TLV has a value other than 8, or the first two octets of its Value field are not 0x030b, the sub-TLV MUST be treated as if it were an unrecognized sub-TLV (see Section 13).

3.5. Embedded Label Handling Sub-TLV (type code 9)

Certain BGP address families (corresponding to particular AFI/SAFI pairs, for example, 1/4, 2/4, 1/128, 2/128) have MPLS labels embedded in their NLRI's. The term "embedded label" is used to refer to the MPLS label that is embedded in an NLRI, and the term "labeled address family" to refer to any AFI/SAFI that has embedded labels.

Some of the tunnel types (for example, VXLAN and NVGRE) that can be specified in the Tunnel Encapsulation attribute have an encapsulation header containing a "Virtual Network" identifier of some sort. The Encapsulation sub-TLVs for these tunnel types may optionally specify a value for the virtual network identifier.

Suppose a Tunnel Encapsulation attribute is attached to an UPDATE of a labeled address family, and it is decided to use a particular tunnel (specified in one of the attribute's TLVs) for transmitting a packet that is being forwarded according to that UPDATE. When forming the encapsulation header for that packet, different deployment scenarios require different handling of the embedded label and/or the virtual network identifier. The Embedded Label Handling sub-TLV can be used to control the placement of the embedded label and/or the virtual network identifier in the encapsulation.

The Embedded Label Handling sub-TLV may be included in any TLV of the Tunnel Encapsulation attribute. If the Tunnel Encapsulation attribute is attached to an UPDATE of a non-labeled address family, then the sub-TLV MUST be disregarded. If the sub-TLV is contained in a TLV whose Tunnel Type does not have a virtual network identifier in

its encapsulation header, the sub-TLV MUST be disregarded. In those cases where the sub-TLV is ignored, it MUST NOT be stripped from the TLV before the route is propagated.

The sub-TLV's Length field always contains the value 1, and its Value field consists of a single octet. The following values are defined:

- 1: The payload will be an MPLS packet with the embedded label at the top of its label stack.
- 2: The embedded label is not carried in the payload, but is carried either in the virtual network identifier field of the encapsulation header, or else is ignored entirely.

If any value other than 1 or 2 is carried, the sub-TLV MUST be considered malformed, according to the procedures of Section 13.

Please see Section 9 for the details of how this sub-TLV is used when it is carried by an UPDATE of a labeled address family.

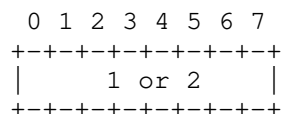


Figure 12: Embedded Label Handling Sub-TLV Value Field

3.6. MPLS Label Stack Sub-TLV (type code 10)

This sub-TLV allows an MPLS label stack ([RFC3032]) to be associated with a particular tunnel.

The length of the sub-TLV is a multiple of 4 octets and the Value field of this sub-TLV is a sequence of MPLS label stack entries. The first entry in the sequence is the "topmost" label, the final entry in the sequence is the "bottommost" label. When this label stack is pushed onto a packet, this ordering MUST be preserved.

Each label stack entry has the following format:

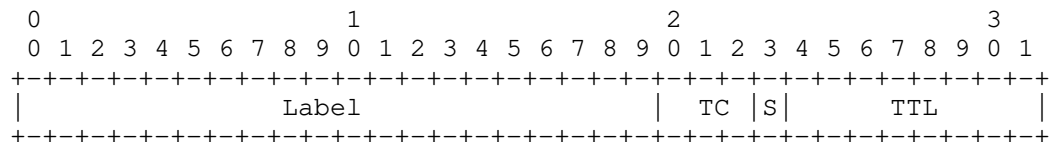


Figure 13: MPLS Label Stack Sub-TLV Value Field

The fields are as defined in [RFC3032], [RFC5462].

If a packet is to be sent through the tunnel identified in a particular TLV, and if that TLV contains an MPLS Label Stack sub-TLV, then the label stack appearing in the sub-TLV MUST be pushed onto the packet before any other labels are pushed onto the packet. (See Section 6 for further discussion.)

In particular, if the Tunnel Encapsulation attribute is attached to a BGP UPDATE of a labeled address family, the contents of the MPLS Label Stack sub-TLV MUST be pushed onto the packet before the label embedded in the NLRI is pushed onto the packet.

If the MPLS Label Stack sub-TLV is included in a TLV identifying a Tunnel Type that uses virtual network identifiers (see Section 9), the contents of the MPLS Label Stack sub-TLV MUST be pushed onto the packet before the procedures of Section 9 are applied.

The number of label stack entries in the sub-TLV MUST be determined from the sub-TLV length field. Thus it is not necessary to set the S bit in any of the label stack entries of the sub-TLV, and the setting of the S bit is ignored when parsing the sub-TLV. When the label stack entries are pushed onto a packet that already has a label stack, the S bits of all the entries being pushed MUST be cleared. When the label stack entries are pushed onto a packet that does not already have a label stack, the S bit of the bottommost label stack entry MUST be set, and the S bit of all the other label stack entries MUST be cleared.

The TC (Traffic Class) field ([RFC3270], [RFC5129]) of each label stack entry SHOULD be set to 0, unless changed by policy at the originator of the sub-TLV. When pushing the label stack onto a packet, the TC of each label stack SHOULD be preserved, unless local policy results in a modification.

The TTL (Time to Live) field of each label stack entry SHOULD be set to 255, unless changed to some other non-zero value by policy at the originator of the sub-TLV. When pushing the label stack onto a packet, the TTL of each label stack entry SHOULD be preserved, unless local policy results in a modification to some other non-zero value. If any label stack entry in the sub-TLV has a TTL value of zero, the router that is pushing the stack on a packet MUST change the value to a non-zero value, either 255 or some other value as determined by policy as discussed above.

Note that this sub-TLV can appear within a TLV identifying any type of tunnel, not just within a TLV identifying an MPLS tunnel. However, if this sub-TLV appears within a TLV identifying an MPLS

tunnel (or an MPLS-in-X tunnel), this sub-TLV plays the same role that would be played by an MPLS Encapsulation sub-TLV. Therefore, an MPLS Encapsulation sub-TLV is not defined.

Although this specification does not supply detailed instructions for validating the received label stack, implementations might impose restrictions on the label stack they can support. If an invalid or unsupported label stack is received, the tunnel MAY be treated as not feasible according to the procedures of Section 6.

3.7. Prefix-SID Sub-TLV (type code 11)

[RFC8669] defines a BGP Path attribute known as the "Prefix-SID Attribute". This attribute is defined to contain a sequence of one or more TLVs, where each TLV is either a "Label-Index" TLV, or an "Originator SRGB (Source Routing Global Block)" TLV.

This document defines a Prefix-SID sub-TLV. The Value field of the Prefix-SID sub-TLV can be set to any permitted value of the Value field of a BGP Prefix-SID attribute [RFC8669].

[RFC8669] only defines behavior when the Prefix-SID Attribute is attached to routes of type IPv4/IPv6 Labeled Unicast ([RFC4760], [RFC8277]), and it only defines values of the Prefix-SID Attribute for those cases. Therefore, similar limitations exist for the Prefix-SID sub-TLV: it SHOULD only be included in a BGP UPDATE message for one of the address families defined in [RFC8669]. If included in a BGP UPDATE for any other address family then it MUST be ignored.

The Prefix-SID sub-TLV can occur in a TLV identifying any type of tunnel. If an Originator SRGB is specified in the sub-TLV, that SRGB MUST be interpreted to be the SRGB used by the tunnel's egress endpoint. The Label-Index, if present, is the Segment Routing SID that the tunnel's egress endpoint uses to represent the prefix appearing in the NLRI field of the BGP UPDATE to which the Tunnel Encapsulation attribute is attached.

If a Label-Index is present in the Prefix-SID sub-TLV, then when a packet is sent through the tunnel identified by the TLV, if that tunnel is from a labeled address family, the corresponding MPLS label MUST be pushed on the packet's label stack. The corresponding MPLS label is computed from the Label-Index value and the SRGB of the route's originator, as specified in section 4.1 of [RFC8669].

The corresponding MPLS label is pushed on after the processing of the MPLS Label Stack sub-TLV, if present, as specified in Section 3.6. It is pushed on before any other labels (for example, a label

embedded in UPDATE's NLRI, or a label determined by the procedures of Section 9), are pushed on the stack.

The Prefix-SID sub-TLV has slightly different semantics than the Prefix-SID attribute. When the Prefix-SID attribute is attached to a given route, the BGP speaker that originally attached the attribute is expected to be in the same Segment Routing domain as the BGP speakers who receive the route with the attached attribute. The Label-Index tells the receiving BGP speakers what the prefix-SID is for the advertised prefix in that Segment Routing domain. When the Prefix-SID sub-TLV is used, there is no implication that the prefix-SID for the advertised prefix is the same in the Segment Routing domains of the BGP speaker that originated the sub-TLV and the BGP speaker that received it.

4. Extended Communities Related to the Tunnel Encapsulation Attribute

4.1. Encapsulation Extended Community

The Encapsulation Extended Community is a Transitive Opaque Extended Community.

The Encapsulation Extended Community encoding is as shown below

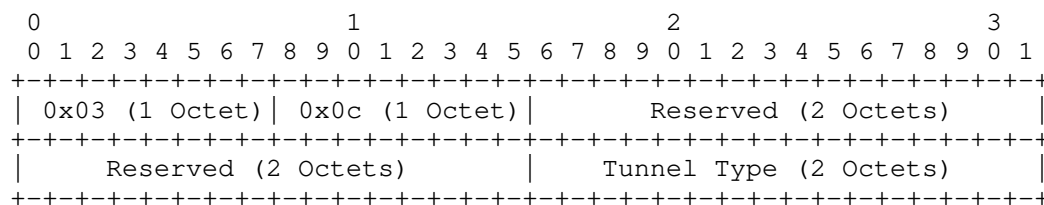


Figure 14: Encapsulation Extended Community

The value of the high-order octet of the extended type field is 0x03, which indicates it's transitive. The value of the low-order octet of the extended type field is 0x0c.

The last two octets of the Value field encode a tunnel type.

This Extended Community may be attached to a route of any AFI/SAFI to which the Tunnel Encapsulation attribute may be attached. Each such Extended Community identifies a particular Tunnel Type, its semantics are the same as semantics of a Tunnel Encapsulation attribute Tunnel TLV for which the following three conditions all hold:

1. it identifies the same Tunnel Type,

2. it has a Tunnel Egress Endpoint sub-TLV for which one of the following two conditions holds:
 - A. its "Address Family" subfield contains zero, or
 - B. its "Address" subfield contains the address of the next hop field of the route to which the Tunnel Encapsulation attribute is attached
3. it has no other sub-TLVs.

Such a Tunnel TLV is called a "barebones" Tunnel TLV.

The Encapsulation Extended Community was first defined in [RFC5512]. While it provides only a small subset of the functionality of the Tunnel Encapsulation attribute, it is used in a number of deployed applications, and is still needed for backwards compatibility. In situations where a tunnel could be encoded using a barebones TLV, it MUST be encoded using the corresponding Encapsulation Extended Community. Notwithstanding, an implementation MUST be prepared to process a tunnel received encoded as a barebones TLV.

Note that for tunnel types of the form "X-in-Y", for example, MPLS-in-GRE, the Encapsulation Extended Community implies that only packets of the specified payload type "X" are to be carried through the tunnel of type "Y". Packets with other payload types MUST NOT be carried through such tunnels. See also Section 2.

In the remainder of this specification, when a route is referred to as containing a Tunnel Encapsulation attribute with a TLV identifying a particular Tunnel Type, it implicitly includes the case where the route contains a Tunnel Encapsulation Extended Community identifying that Tunnel Type.

4.2. Router's MAC Extended Community

[I-D.ietf-bess-evpn-inter-subnet-forwarding] defines a Router's MAC Extended Community. This Extended Community, as its name implies, carries the MAC address of the advertising router. Since the VXLAN and NVGRE Encapsulation Sub-TLVs can also optionally carry a router's MAC, a conflict can arise if both the Router's MAC Extended Community and such an Encapsulation Sub-TLV are present at the same time but have different values. In case of such a conflict, the information in the Router's MAC Extended Community MUST be used.

4.3. Color Extended Community

The Color Extended Community is a Transitive Opaque Extended Community with the following encoding:

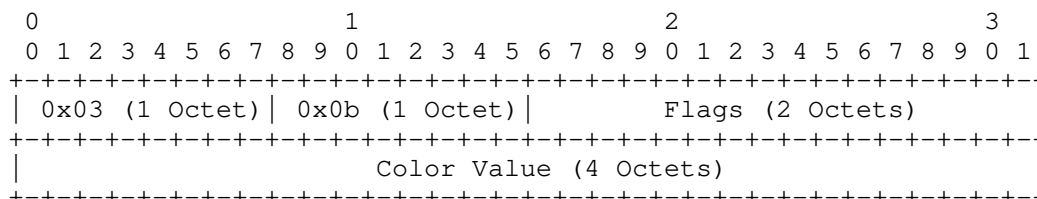


Figure 15: Color Extended Community

The value of the high-order octet of the extended type field is 0x03, which indicates it is transitive. The value of the low-order octet of the extended type field for this community is 0x0b. The color value is user defined and configured locally. No flags are defined in this document; this field MUST be set to zero by the originator and ignored by the receiver; the value MUST NOT be changed when propagating this Extended Community. The Color Value field is encoded as 4 octet value by the administrator and is outside the scope of this document. For the use of this Extended Community please see Section 8.

5. Special Considerations for IP-in-IP Tunnels

In certain situations with an IP fabric underlay, one could have a tunnel overlay with the tunnel type IP-in-IP. The egress BGP speaker can advertise the IP-in-IP tunnel endpoint address in the Tunnel Egress Endpoint sub-TLV. When the Tunnel type of the TLV is IP-in-IP, it will not have a Virtual Network Identifier. However, the tunnel egress endpoint address can be used in identifying the forwarding table to use for making the forwarding decisions to forward the payload.

6. Semantics and Usage of the Tunnel Encapsulation attribute

The BGP Tunnel Encapsulation attribute MAY be carried in any BGP UPDATE message whose AFI/SAFI is 1/1 (IPv4 Unicast), 2/1 (IPv6 Unicast), 1/4 (IPv4 Labeled Unicast), 2/4 (IPv6 Labeled Unicast), 1/128 (VPN-IPv4 Labeled Unicast), 2/128 (VPN-IPv6 Labeled Unicast), or 25/70 (Ethernet VPN, usually known as EVPN)). Use of the Tunnel Encapsulation attribute in BGP UPDATE messages of other AFI/SAFIs is outside the scope of this document.

There is no significance to the order in which the TLVs occur within the Tunnel Encapsulation attribute. Multiple TLVs may occur for a given Tunnel Type; each such TLV is regarded as describing a different tunnel. (This also applies if the Tunnel Encapsulation Extended Community encoding is used.)

The decision to attach a Tunnel Encapsulation attribute to a given BGP UPDATE is determined by policy. The set of TLVs and sub-TLVs contained in the attribute is also determined by policy.

Suppose that:

- o a given packet P must be forwarded by router R;
- o the path along which P is to be forwarded is determined by BGP UPDATE U;
- o UPDATE U has a Tunnel Encapsulation attribute, containing at least one TLV that identifies a "feasible tunnel" for packet P. A tunnel is considered feasible if it has the following four properties:
 - * The Tunnel Type is supported (that is, router R knows how to set up tunnels of that type, how to create the encapsulation header for tunnels of that type, etc.)
 - * The tunnel is of a type that can be used to carry packet P (for example, an MPLS-in-UDP tunnel would not be a feasible tunnel for carrying an IP packet, unless the IP packet can first be encapsulated in a MPLS packet).
 - * The tunnel is specified in a TLV whose Tunnel Egress Endpoint sub-TLV identifies an IP address that is reachable. The reachability condition is evaluated as per [RFC4271]. If the IP address is reachable via more than one forwarding table, local policy is used to determine which table to use.
 - * There is no local policy that prevents the use of the tunnel.

Then router R MUST send packet P through one of the feasible tunnels identified in the Tunnel Encapsulation attribute of UPDATE U.

If the Tunnel Encapsulation attribute contains several TLVs (that is, if it specifies several feasible tunnels), router R may choose any one of those tunnels, based upon local policy. If any Tunnel TLV contains one or more Color sub-TLVs (Section 3.4.2) and/or the Protocol Type sub-TLV (Section 3.4.1), the choice of tunnel may be influenced by these sub-TLVs. Many other factors, for example

minimization of encapsulation header overhead, could also be used to influence selection.

The reachability to the address of the egress endpoint of the tunnel may change over time, directly impacting the feasibility of the tunnel. A tunnel that is not feasible at some moment, may become feasible at a later time when its egress endpoint address is reachable. The router may start using the newly feasible tunnel instead of an existing one. How this decision is made is outside the scope of this document.

Once it is determined to send a packet through the tunnel specified in a particular Tunnel TLV of a particular Tunnel Encapsulation attribute, then the tunnel's egress endpoint address is the IP address contained in the Tunnel Egress Endpoint sub-TLV. If the Tunnel TLV contains a Tunnel Egress Endpoint sub-TLV whose Value field is all zeroes, then the tunnel's egress endpoint is the address of the Next Hop of the BGP Update containing the Tunnel Encapsulation attribute. The address of the tunnel egress endpoint generally appears in a "destination address" field of the encapsulation.

The full set of procedures for sending a packet through a particular Tunnel Type to a particular tunnel egress endpoint depends upon the tunnel type, and is outside the scope of this document. Note that some tunnel types may require the execution of an explicit tunnel setup protocol before they can be used for carrying data. Other tunnel types may not require any tunnel setup protocol.

Sending a packet through a tunnel always requires that the packet be encapsulated, with an encapsulation header that is appropriate for the Tunnel Type. The contents of the tunnel encapsulation header may be influenced by the Encapsulation sub-TLV. If there is no Encapsulation sub-TLV present, the router transmitting the packet through the tunnel must have a priori knowledge (for example, by provisioning) of how to fill in the various fields in the encapsulation header.

A Tunnel Encapsulation attribute may contain several TLVs that all specify the same Tunnel Type. Each TLV should be considered as specifying a different tunnel. Two tunnels of the same type may have different Tunnel Egress Endpoint sub-TLVs, different Encapsulation sub-TLVs, etc. Choosing between two such tunnels is a matter of local policy.

Once router R has decided to send packet P through a particular tunnel, it encapsulates packet P appropriately and then forwards it according to the route that leads to the tunnel's egress endpoint. This route may itself be a BGP route with a Tunnel Encapsulation

attribute. If so, the encapsulated packet is treated as the payload and is encapsulated according to the Tunnel Encapsulation attribute of that route. That is, tunnels may be "stacked".

Notwithstanding anything said in this document, a BGP speaker MAY have local policy that influences the choice of tunnel, and the way the encapsulation is formed. A BGP speaker MAY also have a local policy that tells it to ignore the Tunnel Encapsulation attribute entirely or in part. Of course, interoperability issues must be considered when such policies are put into place.

See also Section 13, which provides further specification regarding validation and exception cases.

7. Routing Considerations

7.1. Impact on the BGP Decision Process

The presence of the Tunnel Encapsulation attribute affects the BGP best route selection algorithm. If a route includes the Tunnel Encapsulation attribute, and if that attribute includes no tunnel which is feasible, then that route MUST NOT be considered resolvable for the purposes of Route Resolvability Condition [RFC4271] Section 9.1.2.1.

7.2. Looping, Mutual Recursion, Etc.

Consider a packet destined for address X. Suppose a BGP UPDATE for address prefix X carries a Tunnel Encapsulation attribute that specifies a tunnel egress endpoint of Y, and suppose that a BGP UPDATE for address prefix Y carries a Tunnel Encapsulation attribute that specifies a tunnel egress endpoint of X. It is easy to see that this can have no good outcome. [RFC4271] describes an analogous case as mutually recursive routes.

This could happen as a result of misconfiguration, either accidental or intentional. It could also happen if the Tunnel Encapsulation attribute were altered by a malicious agent. Implementations should be aware that such an attack will result in unresolvable BGP routes due to the mutually recursive relationship. This document does not specify a maximum number of recursions; that is an implementation-specific matter.

Improper setting (or malicious altering) of the Tunnel Encapsulation attribute could also cause data packets to loop. Suppose a BGP UPDATE for address prefix X carries a Tunnel Encapsulation attribute that specifies a tunnel egress endpoint of Y. Suppose router R receives and processes the advertisement. When router R receives a

packet destined for X, it will apply the encapsulation and send the encapsulated packet to Y. Y will decapsulate the packet and forward it further. If Y is further away from X than is router R, it is possible that the path from Y to X will traverse R. This would cause a long-lasting routing loop. The control plane itself cannot detect this situation, though a TTL field in the payload packets would prevent any given packet from looping infinitely.

During the deployment of techniques as described in this document, operators are encouraged to avoid mutually recursive route and/or tunnel dependencies. There is greater potential for such scenarios to arise when the tunnel egress endpoint for a given prefix differs from the address of the next hop for that prefix.

8. Recursive Next Hop Resolution

Suppose that:

- o a given packet P must be forwarded by router R1;
- o the path along which P is to be forwarded is determined by BGP UPDATE U1;
- o UPDATE U1 does not have a Tunnel Encapsulation attribute;
- o the address of the next hop of UPDATE U1 is router R2;
- o the best route to router R2 is a BGP route that was advertised in UPDATE U2;
- o UPDATE U2 has a Tunnel Encapsulation attribute.

Then packet P MUST be sent through one of the tunnels identified in the Tunnel Encapsulation attribute of UPDATE U2. See Section 6 for further details.

However, suppose that one of the TLVs in U2's Tunnel Encapsulation attribute contains one or more Color Sub-TLVs. In that case, packet P MUST NOT be sent through the tunnel contained in that TLV, unless U1 is carrying a Color Extended Community that is identified in one of U2's Color Sub-TLVs.

The procedures in this section presuppose that U1's address of the next hop resolves to a BGP route, and that U2's next hop resolves (perhaps after further recursion) to a non-BGP route.

9. Use of Virtual Network Identifiers and Embedded Labels when Imposing a Tunnel Encapsulation

If the TLV specifying a tunnel contains an MPLS Label Stack sub-TLV, then when sending a packet through that tunnel, the procedures of Section 3.6 are applied before the procedures of this section.

If the TLV specifying a tunnel contains a Prefix-SID sub-TLV, the procedures of Section 3.7 are applied before the procedures of this section. If the TLV also contains an MPLS Label Stack sub-TLV, the procedures of Section 3.6 are applied before the procedures of Section 3.7.

9.1. Tunnel Types without a Virtual Network Identifier Field

If a Tunnel Encapsulation attribute is attached to an UPDATE of a labeled address family, there will be one or more labels specified in the UPDATE's NLRI. When a packet is sent through a tunnel specified in one of the attribute's TLVs, and that tunnel type does not contain a virtual network identifier field, the label or labels from the NLRI are pushed on the packet's label stack. The resulting MPLS packet is then further encapsulated, as specified by the TLV.

9.2. Tunnel Types with a Virtual Network Identifier Field

Two of the tunnel types that can be specified in a Tunnel Encapsulation TLV have virtual network identifier fields in their encapsulation headers. In the VXLAN encapsulation, this field is called the VNI (VXLAN Network Identifier) field; in the NVGRE encapsulation, this field is called the VSID (Virtual Subnet Identifier) field.

When one of these tunnel encapsulations is imposed on a packet, the setting of the virtual network identifier field in the encapsulation header depends upon the contents of the Encapsulation sub-TLV (if one is present). When the Tunnel Encapsulation attribute is being carried in a BGP UPDATE of a labeled address family, the setting of the virtual network identifier field also depends upon the contents of the Embedded Label Handling sub-TLV (if present).

This section specifies the procedures for choosing the value to set in the virtual network identifier field of the encapsulation header. These procedures apply only when the Tunnel Type is VXLAN or NVGRE.

9.2.1. Unlabeled Address Families

This sub-section applies when:

- o the Tunnel Encapsulation attribute is carried in a BGP UPDATE of an unlabeled address family, and
- o at least one of the attribute's TLVs identifies a Tunnel Type that uses a virtual network identifier, and
- o it has been determined to send a packet through one of those tunnels.

If the TLV identifying the tunnel contains an Encapsulation sub-TLV whose V bit is set, the virtual network identifier field of the encapsulation header is set to the value of the virtual network identifier field of the Encapsulation sub-TLV.

Otherwise, the virtual network identifier field of the encapsulation header is set to a configured value; if there is no configured value, the tunnel cannot be used.

9.2.2. Labeled Address Families

This sub-section applies when:

- o the Tunnel Encapsulation attribute is carried in a BGP UPDATE of a labeled address family, and
- o at least one of the attribute's TLVs identifies a Tunnel Type that uses a virtual network identifier, and
- o it has been determined to send a packet through one of those tunnels.

9.2.2.1. When a Valid VNI has been Signaled

If the TLV identifying the tunnel contains an Encapsulation sub-TLV whose V bit is set, the virtual network identifier field of the encapsulation header is set to the value of the virtual network identifier field of the Encapsulation sub-TLV. However, the Embedded Label Handling sub-TLV will determine label processing as described below.

- o If the TLV contains an Embedded Label Handling sub-TLV whose value is 1, the embedded label (from the NLRI of the route that is carrying the Tunnel Encapsulation attribute) appears at the top of the MPLS label stack in the encapsulation payload.

- o If the TLV does not contain an Embedded Label Handling sub-TLV, or it contains an Embedded Label Handling sub-TLV whose value is 2, the embedded label is ignored entirely.

9.2.2.2. When a Valid VNI has not been Signaled

If the TLV identifying the tunnel does not contain an Encapsulation sub-TLV whose V bit is set, the virtual network identifier field of the encapsulation header is set as follows:

- o If the TLV contains an Embedded Label Handling sub-TLV whose value is 1, then the virtual network identifier field of the encapsulation header is set to a configured value.

If there is no configured value, the tunnel cannot be used.

The embedded label (from the NLRI of the route that is carrying the Tunnel Encapsulation attribute) appears at the top of the MPLS label stack in the encapsulation payload.

- o If the TLV does not contain an Embedded Label Handling sub-TLV, or if it contains an Embedded Label Handling sub-TLV whose value is 2, the embedded label is copied into the lower 3 octets of the virtual network identifier field of the encapsulation header.

In this case, the payload may or may not contain an MPLS label stack, depending upon other factors. If the payload does contain an MPLS label stack, the embedded label does not appear in that stack.

10. Applicability Restrictions

In a given UPDATE of a labeled address family, the label embedded in the NLRI is generally a label that is meaningful only to the router represented by the address of the next hop. Certain of the procedures of Section 9.2.2.1 or Section 9.2.2.2 cause the embedded label to be carried by a data packet to the router whose address appears in the Tunnel Egress Endpoint sub-TLV. If the Tunnel Egress Endpoint sub-TLV does not identify the same router represented by the address of the next hop, sending the packet through the tunnel may cause the label to be misinterpreted at the tunnel's egress endpoint. This may cause misdelivery of the packet. Avoidance of this unfortunate outcome is a matter of network planning and design, and is outside the scope of this document.

Note that if the Tunnel Encapsulation attribute is attached to a VPN-IP route [RFC4364], and if Inter-AS "option b" (see section 10 of [RFC4364]) is being used, and if the Tunnel Egress Endpoint sub-TLV

contains an IP address that is not in same AS as the router receiving the route, it is very likely that the embedded label has been changed. Therefore use of the Tunnel Encapsulation attribute in an "Inter-AS option b" scenario is not recommended.

Other documents may define other ways to signal tunnel information in BGP. For example, [RFC6514] defines the "P-Multicast Service Interface Tunnel" (PMSI Tunnel) attribute. In this specification, we do not consider the effects of advertising the Tunnel Encapsulation Attribute in conjunction with other forms of signaling tunnels. Any document specifying such joint use MUST provide details as to how interactions should be handled.

11. Scoping

The Tunnel Encapsulation attribute is defined as a transitive attribute, so that it may be passed along by BGP speakers that do not recognize it. However the Tunnel Encapsulation attribute MUST be used only within a well-defined scope, for example, within a set of Autonomous Systems that belong to a single administrative entity. If the attribute is distributed beyond its intended scope, packets may be sent through tunnels in a manner that is not intended.

To prevent the Tunnel Encapsulation attribute from being distributed beyond its intended scope, any BGP speaker that understands the attribute MUST be able to filter the attribute from incoming BGP UPDATE messages. When the attribute is filtered from an incoming UPDATE, the attribute is neither processed nor distributed. This filtering SHOULD be possible on a per-BGP-session basis; finer granularities (for example, per route and/or per attribute TLV) MAY be supported. For each external BGP (EBGP) session, filtering of the attribute on incoming UPDATES MUST be enabled by default.

In addition, any BGP speaker that understands the attribute MUST be able to filter the attribute from outgoing BGP UPDATE messages. This filtering SHOULD be possible on a per-BGP-session basis. For each EBGP session, filtering of the attribute on outgoing UPDATES MUST be enabled by default.

Since the Tunnel Encapsulation Extended Community provides a subset of the functionality of the Tunnel Encapsulation attribute, these considerations apply equally in its case: any BGP speaker that understands it MUST be able to filter it from incoming BGP UPDATE messages, it MUST be possible to filter the Tunnel Encapsulation Extended Community from outgoing messages, and in both cases this filtering MUST be enabled by default for EBGP sessions.

12. Operational Considerations

A potential operational difficulty arises when tunnels are used, if the size of packets entering the tunnel exceeds the maximum transmission unit (MTU) the tunnel is capable of supporting. This difficulty can be exacerbated by stacking multiple tunnels, since each stacked tunnel header further reduces the supportable MTU. This issue is long-standing and well-known. The tunnel signaling provided in this specification does nothing to address this issue, nor to aggravate it (except insofar as it may further increase the popularity of tunneling).

13. Validation and Error Handling

The Tunnel Encapsulation attribute is a sequence of TLVs, each of which is a sequence of sub-TLVs. The final octet of a TLV is determined by its length field. Similarly, the final octet of a sub-TLV is determined by its length field. The final octet of a TLV MUST also be the final octet of its final sub-TLV. If this is not the case, the TLV MUST be considered to be malformed, and the "Treat-as-withdraw" procedure of [RFC7606] is applied.

If a Tunnel Encapsulation attribute does not have any valid TLVs, or it does not have the transitive bit set, the "Treat-as-withdraw" procedure of [RFC7606] is applied.

If a Tunnel Encapsulation attribute can be parsed correctly, but contains a TLV whose Tunnel Type is not recognized by a particular BGP speaker, that BGP speaker MUST NOT consider the attribute to be malformed. Rather, it MUST interpret the attribute as if that TLV had not been present. If the route carrying the Tunnel Encapsulation attribute is propagated with the attribute, the unrecognized TLV MUST remain in the attribute.

The following sub-TLVs defined in this document MUST NOT occur more than once in a given Tunnel TLV: Tunnel Egress Endpoint (discussed below), Encapsulation, DS, UDP Destination Port, Embedded Label Handling, MPLS Label Stack, Prefix-SID. If a Tunnel TLV has more than one of any of these sub-TLVs, all but the first occurrence of each such sub-TLV type MUST be disregarded. However, the Tunnel TLV containing them MUST NOT be considered to be malformed, and all the sub-TLVs MUST be propagated if the route carrying the Tunnel Encapsulation attribute is propagated.

The following sub-TLVs defined in this document may appear zero or more times in a given Tunnel TLV: Protocol Type, Color. Each occurrence of such sub-TLVs is meaningful. For example, the Color

sub-TLV may appear multiple times to assign multiple colors to a tunnel.

If a TLV of a Tunnel Encapsulation attribute contains a sub-TLV that is not recognized by a particular BGP speaker, the BGP speaker MUST process that TLV as if the unrecognized sub-TLV had not been present. If the route carrying the Tunnel Encapsulation attribute is propagated with the attribute, the unrecognized sub-TLV MUST remain in the attribute.

In general, if a TLV contains a sub-TLV that is malformed, the sub-TLV MUST be treated as if it were an unrecognized sub-TLV. There is one exception to this rule -- if a TLV contains a malformed Tunnel Egress Endpoint sub-TLV (as defined in Section 3.1), the entire TLV MUST be ignored, and MUST be removed from the Tunnel Encapsulation attribute before the route carrying that attribute is distributed.

Within a Tunnel Encapsulation attribute that is carried by a BGP UPDATE whose AFI/SAFI is one of those explicitly listed in the second paragraph of Section 6, a TLV that does not contain exactly one Tunnel Egress Endpoint sub-TLV MUST be treated as if it contained a malformed Tunnel Egress Endpoint sub-TLV.

A TLV identifying a particular Tunnel Type may contain a sub-TLV that is meaningless for that Tunnel Type. For example, perhaps the TLV contains a UDP Destination Port sub-TLV, but the identified tunnel type does not use UDP encapsulation at all, or a tunnel of the form "X-in-Y" contains a Protocol Type sub-TLV that specifies something other than "X". Sub-TLVs of this sort MUST be disregarded. That is, they MUST NOT affect the creation of the encapsulation header. However, the sub-TLV MUST NOT be considered to be malformed, and MUST NOT be removed from the TLV before the route carrying the Tunnel Encapsulation attribute is distributed. An implementation MAY log a message when it encounters such a sub-TLV.

14. IANA Considerations

This document makes the following requests of IANA. (All registration procedures listed below are per their definitions in [RFC8126].)

14.1. Obsoleting RFC 5512

Because this document obsoletes RFC 5512, change all registration information that references [RFC5512] to instead reference this document.

14.2. Obsoleting Code Points Assigned by RFCs 5566

Since this document obsoletes RFC 5566, the code points assigned by that RFC are similarly obsoleted. Specifically, the following code points should be marked as deprecated.

In the "BGP Tunnel Encapsulation Attribute Tunnel Types" registry:

| Value | Name |
|-------|---|
| 3 | Transmit tunnel endpoint |
| 4 | IPsec in Tunnel-mode |
| 5 | IP in IP tunnel with IPsec Transport Mode |
| 6 | MPLS-in-IP tunnel with IPsec Transport Mode |

And in the "BGP Tunnel Encapsulation Attribute Sub-TLVs" registry:

| Value | Name |
|-------|----------------------------|
| 3 | IPsec Tunnel Authenticator |

14.3. BGP Tunnel Encapsulation Parameters Grouping

Create a new registry grouping, to be named "BGP Tunnel Encapsulation Parameters".

14.4. BGP Tunnel Encapsulation Attribute Tunnel Types

Relocate the "BGP Tunnel Encapsulation Attribute Tunnel Types" registry to be under the "BGP Tunnel Encapsulation Parameters" grouping.

14.5. Subsequent Address Family Identifiers

Modify the "Subsequent Address Family Identifiers" registry to indicate that the Encapsulation SAFI (value 7) is obsoleted. This document should be the reference.

14.6. BGP Tunnel Encapsulation Attribute Sub-TLVs

Relocate the "BGP Tunnel Encapsulation Attribute Sub-TLVs" registry to be under the "BGP Tunnel Encapsulation Parameters" grouping.

Add the following note to the registry:

If the Sub-TLV Type is in the range from 0 to 127 inclusive, the Sub-TLV Length field contains one octet. If the Sub-TLV Type is in the range from 128-255 inclusive, the Sub-TLV Length field contains two octets.

Change the registration policy of the registry to the following:

| Value(s) | Registration Procedure |
|----------|-------------------------|
| 0 | Reserved |
| 1-63 | Standards Action |
| 64-125 | First Come First Served |
| 126-127 | Experimental Use |
| 128-191 | Standards Action |
| 192-252 | First Come First Served |
| 253-254 | Experimental Use |
| 255 | Reserved |

Rename the following entries within the registry:

| Value | Old Name | New Name |
|-------|-----------------|------------------------|
| 6 | Remote Endpoint | Tunnel Egress Endpoint |
| 7 | IPv4 DS Field | DS Field |

14.7. Flags Field of VXLAN Encapsulation sub-TLV

Create a registry named "Flags Field of VXLAN Encapsulation sub-TLV" under the "BGP Tunnel Encapsulation Parameters" grouping. The registration policy for this registry is "Standards Action". The minimum possible value is 0, the maximum is 7.

The initial values for this new registry are indicated below.

| Bit Position | Description | Reference |
|--------------|-----------------|-----------------|
| 0 | V (VN-ID) | (this document) |
| 1 | M (MAC Address) | (this document) |

14.8. Flags Field of NVGRE Encapsulation sub-TLV

Create a registry named "Flags Field of NVGRE Encapsulation sub-TLV" under the "BGP Tunnel Encapsulation Parameters" grouping. The registration policy for this registry is "Standards Action". The minimum possible value is 0, the maximum is 7.

The initial values for this new registry are indicated below.

| Bit Position | Description | Reference |
|--------------|-----------------|-----------------|
| 0 | V (VN-ID) | (this document) |
| 1 | M (MAC Address) | (this document) |

14.9. Embedded Label Handling sub-TLV

Create a registry named "Embedded Label Handling sub-TLV" under the "BGP Tunnel Encapsulation Parameters" grouping. The registration policy for this registry is "Standards Action". The minimum possible value is 0, the maximum is 255.

The initial values for this new registry are indicated below.

| Value | Description | Reference |
|-------|-------------------------------------|-----------------|
| 0 | Reserved | (this document) |
| 1 | Payload of MPLS with embedded label | (this document) |
| 2 | no embedded label in payload | (this document) |

14.10. Color Extended Community Flags

Create a registry named "Color Extended Community Flags" under the "BGP Tunnel Encapsulation Parameters" grouping. The registration policy for this registry is "Standards Action". The minimum possible value is 0, the maximum is 15.

No initial values are to be registered. The format of the registry is shown below.

| Bit Position | Description | Reference |
|--------------|-------------|-----------|
|--------------|-------------|-----------|

15. Security Considerations

As Section 11 discusses, it is intended that the Tunnel Encapsulation attribute be used only within a well-defined scope, for example, within a set of Autonomous Systems that belong to a single administrative entity. As long as the filtering mechanisms discussed in that section are applied diligently, an attacker outside the scope would not be able to use the Tunnel Encapsulation attribute in an attack. This leaves open the questions of attackers within the scope (for example, a compromised router) and failures in filtering that allow an external attack to succeed.

As [RFC4272] discusses, BGP is vulnerable to traffic diversion attacks. The Tunnel Encapsulation attribute adds a new means by which an attacker could cause traffic to be diverted from its normal path, especially when the Tunnel Egress Endpoint sub-TLV is used. Such an attack would differ from pre-existing vulnerabilities in that traffic could be tunneled to a distant target across intervening network infrastructure, allowing an attack to potentially succeed more easily, since less infrastructure would have to be subverted. Potential consequences include "hijacking" of traffic (insertion of an undesired node in the path allowing for inspection or modification of traffic, or avoidance of security controls) or denial of service (directing traffic to a node that doesn't desire to receive it).

In order to further mitigate the risk of diversion of traffic from its intended destination, Section 3.1.1 provides an optional procedure to check that the destination given in a Tunnel Egress Endpoint sub-TLV is within the AS that was the source of the route. One then has some level of assurance that the tunneled traffic is going to the same destination AS that it would have gone to had the Tunnel Encapsulation attribute not been present. As RFC 4272 discusses, it's possible for an attacker to announce an inaccurate AS_PATH, therefore an attacker with the ability to inject a Tunnel Egress Endpoint sub-TLV could equally craft an AS_PATH that would pass the validation procedures of Section 3.1.1. BGP Origin Validation [RFC6811] and BGPsec [RFC8205] provide means to increase assurance that the origins being validated have not been falsified.

Many tunnels carry traffic that embeds a destination address that comes from a non-global namespace. One example is MPLS VPNs. If a tunnel crosses from one namespace to another, without the necessary translation being performed for the embedded address(es), there exists a risk of misdelivery of traffic. If the traffic contains confidential data that's not otherwise protected (for example, by end-to-end encryption) then confidential information could be revealed. The restriction of applicability of the Tunnel Encapsulation attribute to a well-defined scope limits the likelihood

of this occurring. See the discussion of "option b" in Section 10 for further discussion of one such scenario.

RFC 8402 specifies that "SR domain boundary routers MUST filter any external traffic" ([RFC8402] Section 8.1). For these purposes, traffic introduced into a SR domain using the Prefix-SID sub-TLV lies within the SR domain, even though the prefix-SIDs used by the routers at the two ends of the tunnel may be different, as discussed in Section 3.7. This implies that the duty to filter external traffic extends to all routers participating in such tunnels.

16. Acknowledgments

This document contains text from RFC 5512, authored by Pradosh Mohapatra and Eric Rosen. The authors of the current document wish to thank them for their contribution. RFC 5512 itself built upon prior work by Gargi Nalawade, Ruchi Kapoor, Dan Tappan, David Ward, Scott Wainner, Simon Barber, Lili Wang, and Chris Metz, whom the authors also thank for their contributions. Eric Rosen was the principal author of earlier versions of this document.

The authors wish to thank Lou Berger, Ron Bonica, Martin Djernaes, John Drake, Susan Hares, Satoru Matsushima, Thomas Morin, Dhananjaya Rao, Ravi Singh, Harish Sitaraman, Brian Trammell, Xiaohu Xu, and Zhaohui Zhang for their review, comments, and/or helpful discussions. Alvaro Retana provided an especially comprehensive review.

17. Contributor Addresses

Below is a list of other contributing authors in alphabetical order:

Randy Bush
Internet Initiative Japan
5147 Crystal Springs
Bainbridge Island, Washington 98110
United States

Email: randy@psg.com

Robert Raszuk
Bloomberg LP
731 Lexington Ave
New York City, NY 10022
United States

Email: robert@raszuk.net

Eric C. Rosen

18. References

18.1. 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>>.
- [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>>.
- [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <<https://www.rfc-editor.org/info/rfc2784>>.
- [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>>.
- [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>>.

- [RFC3270] Le Faucheur, F., Wu, L., Davie, B., Davari, S., Vaananen, P., Krishnan, R., Cheval, P., and J. Heinanen, "Multi-Protocol Label Switching (MPLS) Support of Differentiated Services", RFC 3270, DOI 10.17487/RFC3270, May 2002, <<https://www.rfc-editor.org/info/rfc3270>>.
- [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>>.
- [RFC4023] Worster, T., Rekhter, Y., and E. Rosen, Ed., "Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE)", RFC 4023, DOI 10.17487/RFC4023, March 2005, <<https://www.rfc-editor.org/info/rfc4023>>.
- [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>>.
- [RFC5129] Davie, B., Briscoe, B., and J. Tay, "Explicit Congestion Marking in MPLS", RFC 5129, DOI 10.17487/RFC5129, January 2008, <<https://www.rfc-editor.org/info/rfc5129>>.
- [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>>.
- [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>>.
- [RFC6890] Cotton, M., Vegoda, L., Bonica, R., Ed., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC 6890, DOI 10.17487/RFC6890, April 2013, <<https://www.rfc-editor.org/info/rfc6890>>.

- [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>>.
- [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>>.
- [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>>.
- [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>>.
- [RFC8669] Previdi, S., Filsfils, C., Lindem, A., Ed., Sreekantiah, A., and H. Gredler, "Segment Routing Prefix Segment Identifier Extensions for BGP", RFC 8669, DOI 10.17487/RFC8669, December 2019, <<https://www.rfc-editor.org/info/rfc8669>>.

18.2. Informative References

- [Ethertypes] "IANA Ethertype Registry", <<http://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.xhtml>>.
- [I-D.ietf-bess-evpn-inter-subnet-forwarding] Sajassi, A., Salam, S., Thoria, S., Drake, J., and J. Rabadan, "Integrated Routing and Bridging in EVPN", draft-ietf-bess-evpn-inter-subnet-forwarding-11 (work in progress), October 2020.
- [RFC4272] Murphy, S., "BGP Security Vulnerabilities Analysis", RFC 4272, DOI 10.17487/RFC4272, January 2006, <<https://www.rfc-editor.org/info/rfc4272>>.

- [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>>.
- [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>>.
- [RFC5565] Wu, J., Cui, Y., Metz, C., and E. Rosen, "Softwire Mesh Framework", RFC 5565, DOI 10.17487/RFC5565, June 2009, <<https://www.rfc-editor.org/info/rfc5565>>.
- [RFC5566] Berger, L., White, R., and E. Rosen, "BGP IPsec Tunnel Encapsulation Attribute", RFC 5566, DOI 10.17487/RFC5566, June 2009, <<https://www.rfc-editor.org/info/rfc5566>>.
- [RFC5640] Filsfils, C., Mohapatra, P., and C. Pignataro, "Load-Balancing for Mesh Softwires", RFC 5640, DOI 10.17487/RFC5640, August 2009, <<https://www.rfc-editor.org/info/rfc5640>>.
- [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>>.
- [RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <<https://www.rfc-editor.org/info/rfc7510>>.
- [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>>.
- [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>>.
- [RFC8365] Sajassi, A., Ed., Drake, J., Ed., Bitar, N., Shekhar, R., Uttaro, J., and W. Henderickx, "A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)", RFC 8365, DOI 10.17487/RFC8365, March 2018, <<https://www.rfc-editor.org/info/rfc8365>>.

[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>>.

Appendix A. Impact on RFC 8365

[RFC8365] references RFC 5512 for its definition of the BGP Encapsulation Extended Community. That extended community is now defined in this document, in a way consistent with its previous definition.

RFC 8365 talks in Section 6 about the use of the Encapsulation Extended Community to allow Network Virtualization Edge devices (NVEs) to signal their supported encapsulations. We note that with the introduction of this specification, the Tunnel Encapsulation Attribute can also be used for this purpose. For purposes where RFC 8365 talks about "advertising supported encapsulations" (for example, in the second paragraph of Section 6), encapsulations advertised using the Tunnel Encapsulation Attribute should be considered equally with those advertised using the Encapsulation Extended Community.

In particular, a review of Section 8.3.1 of RFC 8365 is called for, to consider whether the introduction of the Tunnel Encapsulation Attribute creates a need for any revisions to the split horizon procedures.

RFC 8365 also refers to a draft version of this specification in the final paragraph of section 5.1.3. That paragraph references Section 8.2.2.2 of the draft. In this version of the document the correct reference would be Section 9.2.2.2. There are no substantive differences between the section in the referenced draft, and that in this document.

Authors' Addresses

Keyur Patel
Arrcus, Inc
2077 Gateway Pl
San Jose, CA 95110
United States

Email: keyur@arrcus.com

Gunter Van de Velde
Nokia
Copernicuslaan 50
Antwerpen 2018
Belgium

Email: gunter.van_de_velde@nokia.com

Srihari R. Sangli
Juniper Networks

Email: ssangli@juniper.net

John Scudder
Juniper Networks

Email: jgs@juniper.net

Inter-Domain Routing
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2020

K. Talaulikar, Ed.
Cisco Systems
H. Gredler
Rtbrick
J. Medved
Cisco Systems, Inc.
S. Previdi
Individual Contributor
A. Farrel
Old Dog Consulting
S. Ray
Individual Contributor
July 8, 2019

Distribution of Link-State and Traffic Engineering Information Using BGP
draft-ketant-idr-rfc7752bis-01

Abstract

In a number of environments, a component external to a network is called upon to perform computations based on the network topology and current state of the connections within the network, including Traffic Engineering (TE) information. This is information typically distributed by IGP routing protocols within the network.

This document describes a mechanism by which link-state and TE information can be collected from networks and shared with external components using the BGP routing protocol. This is achieved using a new BGP Network Layer Reachability Information (NLRI) encoding format. The mechanism is applicable to physical and virtual IGP links. The mechanism described is subject to policy control.

Applications of this technique include Application-Layer Traffic Optimization (ALTO) servers and Path Computation Elements (PCEs).

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 January 9, 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 | 3 |
| 2. Motivation and Applicability | 5 |
| 2.1. MPLS-TE with PCE | 5 |
| 2.2. ALTO Server Network API | 7 |
| 3. BGP Speaker Roles for BGP-LS | 8 |
| 4. Carrying Link-State Information in BGP | 9 |
| 4.1. TLV Format | 9 |
| 4.2. The Link-State NLRI | 10 |
| 4.2.1. Node Descriptors | 15 |
| 4.2.2. Link Descriptors | 19 |
| 4.2.3. Prefix Descriptors | 21 |
| 4.3. The BGP-LS Attribute | 23 |
| 4.3.1. Node Attribute TLVs | 24 |
| 4.3.2. Link Attribute TLVs | 27 |

| | |
|---|----|
| 4.3.3. Prefix Attribute TLVs | 32 |
| 4.4. Private Use | 35 |
| 4.5. BGP Next-Hop Information | 36 |
| 4.6. Inter-AS Links | 36 |
| 4.7. Handling of Unreachable IGP Nodes | 36 |
| 4.8. Router-ID Anchoring Example: ISO Pseudonode | 38 |
| 4.9. Router-ID Anchoring Example: OSPF Pseudonode | 39 |
| 4.10. Router-ID Anchoring Example: OSPFv2 to IS-IS Migration | 40 |
| 5. Link to Path Aggregation | 40 |
| 5.1. Example: No Link Aggregation | 41 |
| 5.2. Example: ASBR to ASBR Path Aggregation | 41 |
| 5.3. Example: Multi-AS Path Aggregation | 42 |
| 6. IANA Considerations | 42 |
| 6.1. Guidance for Designated Experts | 43 |
| 7. Manageability Considerations | 44 |
| 7.1. Operational Considerations | 44 |
| 7.1.1. Operations | 44 |
| 7.1.2. Installation and Initial Setup | 44 |
| 7.1.3. Migration Path | 44 |
| 7.1.4. Requirements on Other Protocols and Functional Components | 44 |
| 7.1.5. Impact on Network Operation | 45 |
| 7.1.6. Verifying Correct Operation | 45 |
| 7.2. Management Considerations | 45 |
| 7.2.1. Management Information | 45 |
| 7.2.2. Fault Management | 45 |
| 7.2.3. Configuration Management | 48 |
| 7.2.4. Accounting Management | 48 |
| 7.2.5. Performance Management | 48 |
| 7.2.6. Security Management | 49 |
| 8. TLV/Sub-TLV Code Points Summary | 49 |
| 9. Security Considerations | 50 |
| 10. Contributors | 51 |
| 11. Acknowledgements | 51 |
| 12. References | 52 |
| 12.1. Normative References | 52 |
| 12.2. Informative References | 54 |
| Appendix A. Changes from RFC 7752 | 56 |
| Authors' Addresses | 57 |

1. Introduction

The contents of a Link-State Database (LSDB) or of an IGP's Traffic Engineering Database (TED) describe only the links and nodes within an IGP area. Some applications, such as end-to-end Traffic Engineering (TE), would benefit from visibility outside one area or Autonomous System (AS) in order to make better decisions.

The IETF has defined the Path Computation Element (PCE) [RFC4655] as a mechanism for achieving the computation of end-to-end TE paths that cross the visibility of more than one TED or that require CPU-intensive or coordinated computations. The IETF has also defined the ALTO server [RFC5693] as an entity that generates an abstracted network topology and provides it to network-aware applications.

Both a PCE and an ALTO server need to gather information about the topologies and capabilities of the network in order to be able to fulfill their function.

This document describes a mechanism by which link-state and TE information can be collected from networks and shared with external components using the BGP routing protocol [RFC4271]. This is achieved using a new BGP Network Layer Reachability Information (NLRI) encoding format. The mechanism is applicable to physical and virtual links. The mechanism described is subject to policy control.

A router maintains one or more databases for storing link-state information about nodes and links in any given area. Link attributes stored in these databases include: local/remote IP addresses, local/remote interface identifiers, link metric and TE metric, link bandwidth, reservable bandwidth, per Class-of-Service (CoS) class reservation state, preemption, and Shared Risk Link Groups (SRLGs). The router's BGP process can retrieve topology from these LSDBs and distribute it to a consumer, either directly or via a peer BGP speaker (typically a dedicated Route Reflector), using the encoding specified in this document.

An illustration of the collection of link-state and TE information and its distribution to consumers is shown in the Figure 1 below.

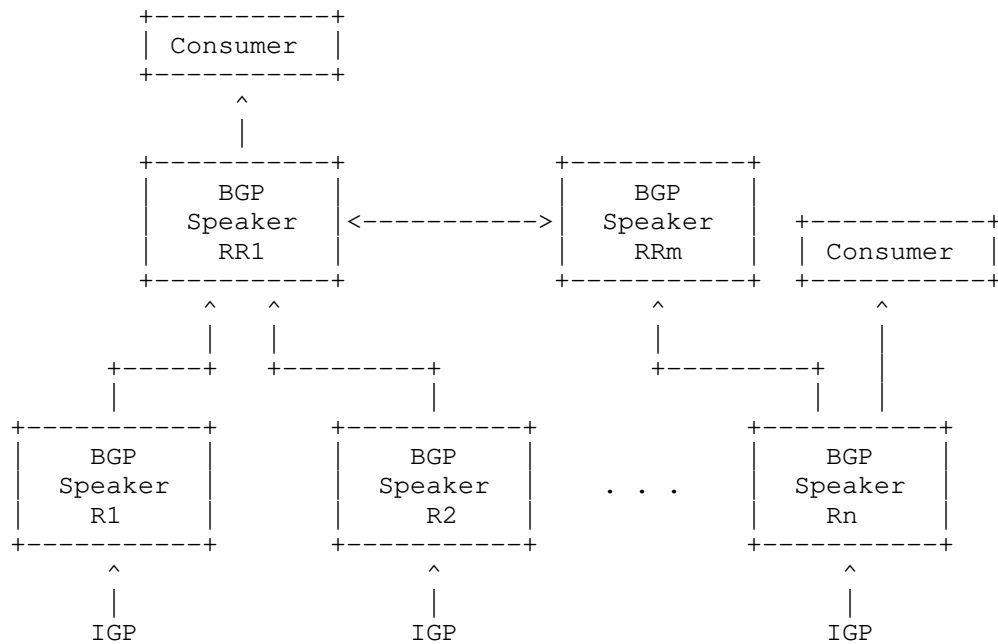


Figure 1: Collection of Link-State and TE Information

A BGP speaker may apply configurable policy to the information that it distributes. Thus, it may distribute the real physical topology from the LSDB or the TED. Alternatively, it may create an abstracted topology, where virtual, aggregated nodes are connected by virtual paths. Aggregated nodes can be created, for example, out of multiple routers in a Point of Presence (POP). Abstracted topology can also be a mix of physical and virtual nodes and physical and virtual links. Furthermore, the BGP speaker can apply policy to determine when information is updated to the consumer so that there is a reduction of information flow from the network to the consumers. Mechanisms through which topologies can be aggregated or virtualized are outside the scope of this document

2. Motivation and Applicability

This section describes use cases from which the requirements can be derived.

2.1. MPLS-TE with PCE

As described in [RFC4655], a PCE can be used to compute MPLS-TE paths within a "domain" (such as an IGP area) or across multiple domains (such as a multi-area AS or multiple ASes).

- o Within a single area, the PCE offers enhanced computational power that may not be available on individual routers, sophisticated policy control and algorithms, and coordination of computation across the whole area.
- o If a router wants to compute a MPLS-TE path across IGP areas, then its own TED lacks visibility of the complete topology. That means that the router cannot determine the end-to-end path and cannot even select the right exit router (Area Border Router (ABR)) for an optimal path. This is an issue for large-scale networks that need to segment their core networks into distinct areas but still want to take advantage of MPLS-TE.

Previous solutions used per-domain path computation [RFC5152]. The source router could only compute the path for the first area because the router only has full topological visibility for the first area along the path, but not for subsequent areas. Per-domain path computation uses a technique called "loose-hop-expansion" [RFC3209] and selects the exit ABR and other ABRs or AS Border Routers (ASBRs) using the IGP-computed shortest path topology for the remainder of the path. This may lead to sub-optimal paths, makes alternate/back-up path computation hard, and might result in no TE path being found when one really does exist.

The PCE presents a computation server that may have visibility into more than one IGP area or AS, or may cooperate with other PCEs to perform distributed path computation. The PCE obviously needs access to the TED for the area(s) it serves, but [RFC4655] does not describe how this is achieved. Many implementations make the PCE a passive participant in the IGP so that it can learn the latest state of the network, but this may be sub-optimal when the network is subject to a high degree of churn or when the PCE is responsible for multiple areas.

The following figure shows how a PCE can get its TED information using the mechanism described in this document.

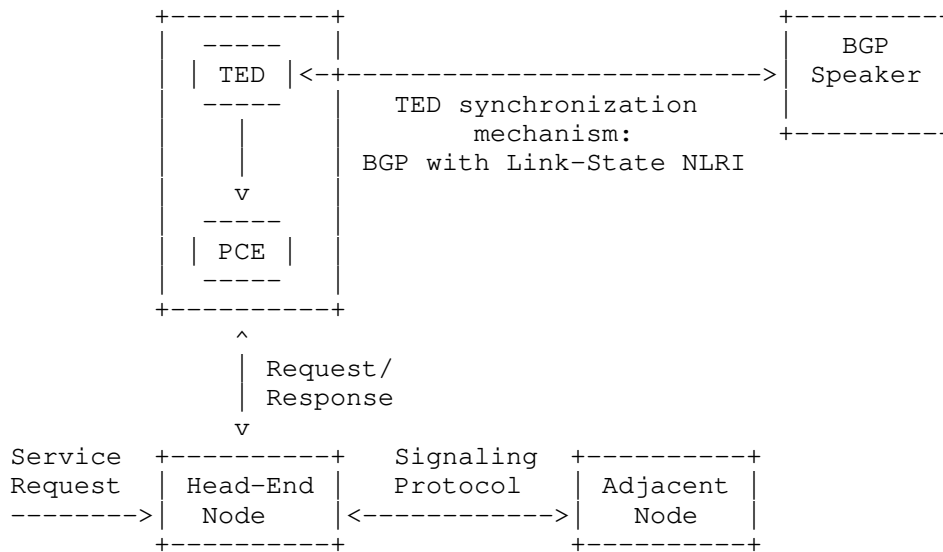


Figure 2: External PCE Node Using a TED Synchronization Mechanism

The mechanism in this document allows the necessary TED information to be collected from the IGP within the network, filtered according to configurable policy, and distributed to the PCE as necessary.

2.2. ALTO Server Network API

An ALTO server [RFC5693] is an entity that generates an abstracted network topology and provides it to network-aware applications over a web-service-based API. Example applications are peer-to-peer (P2P) clients or trackers, or Content Distribution Networks (CDNs). The abstracted network topology comes in the form of two maps: a Network Map that specifies allocation of prefixes to Partition Identifiers (PIDs), and a Cost Map that specifies the cost between PIDs listed in the Network Map. For more details, see [RFC7285].

ALTO abstract network topologies can be auto-generated from the physical topology of the underlying network. The generation would typically be based on policies and rules set by the operator. Both prefix and TE data are required: prefix data is required to generate ALTO Network Maps, and TE (topology) data is required to generate ALTO Cost Maps. Prefix data is carried and originated in BGP, and TE data is originated and carried in an IGP. The mechanism defined in this document provides a single interface through which an ALTO server can retrieve all the necessary prefix and network topology data from the underlying network. Note that an ALTO server can use

other mechanisms to get network data, for example, peering with multiple IGP and BGP speakers.

The following figure shows how an ALTO server can get network topology information from the underlying network using the mechanism described in this document.

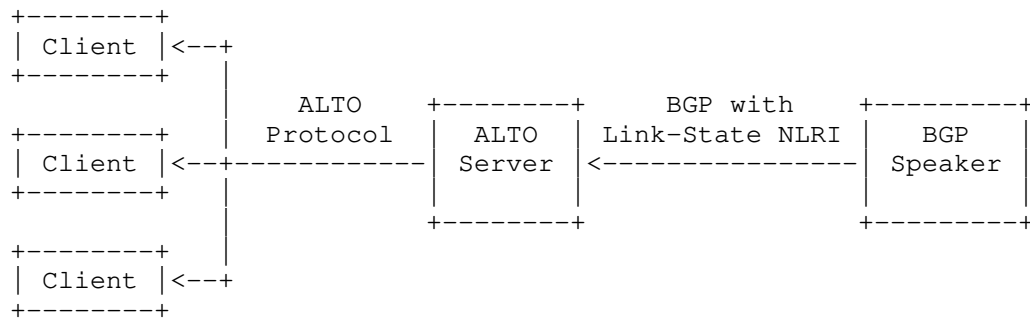


Figure 3: ALTO Server Using Network Topology Information

3. BGP Speaker Roles for BGP-LS

In the illustration shown in Figure 1, the BGP Speakers can be seen playing different roles in the distribution of information using BGP-LS. This section introduces terms that explain the different roles of the BGP Speakers which are then used through the rest of this document.

- o **BGP-LS Producer:** The BGP Speakers R1, R2, ... Rn, originate link-state information from their underlying link-state IGP protocols into BGP-LS. If R1 and R2 are in the same IGP area, then likely they are originating the same link-state information into BGP-LS. R1 may also source information from sources other than IGP, e.g. its local node information. The term BGP-LS Producer refers to the BGP Speaker that is originating link-state information into BGP.
- o **BGP-LS Consumer:** The BGP Speakers RR1 and Rn are handing off the BGP-LS information that they have collected to a consumer application. The BGP protocol implementation and the consumer application may be on the same or different nodes. The term BGP-LS Consumer refers to the consumer application/process and not the BGP Speaker. This document only covers the BGP implementation. The consumer application and the design of interface between BGP and consumer application may be implementation specific and outside the scope of this document.

- o BGP-LS Propagator: The BGP Speaker RRm propagates the BGP-LS information between the BGP Speaker Rn and the BGP Speaker RR1. The BGP implementation on RRm is doing the propagation of BGP-LS updates and performing BGP best path calculations. Similarly, the BGP Speaker RR1 is receiving BGP-LS information from R1, R2 and RRm and propagating the information to the BGP-LS Consumer after performing BGP best path calculations. The term BGP-LS Propagator refers to the BGP Speaker that is performing BGP protocol processing on the link-state information.

The above roles are not mutually exclusive. The same BGP Speaker may be the producer for some link-state information and propagator for some other link-state information while also providing this information to a consumer application. Nothing precludes a BGP implementation performing some of the validation and processing on behalf of the BGP-LS Consumer as long as it does not impact the semantics of its role as BGP-LS Propagator as described in this document.

The rest of this document refers to the role when describing procedures that are specific to that role. When the role is not specified, then the said procedure applies to all BGP Speakers.

4. Carrying Link-State Information in BGP

This specification contains two parts: definition of a new BGP NLRI that describes links, nodes, and prefixes comprising IGP link-state information and definition of a new BGP path attribute (BGP-LS Attribute) that carries link, node, and prefix properties and attributes, such as the link and prefix metric or auxiliary Router-IDs of nodes, etc.

It is desirable to keep the dependencies on the protocol source of this attribute to a minimum and represent any content in an IGP-neutral way, such that applications that want to learn about a link-state topology do not need to know about any OSPF or IS-IS protocol specifics.

This section mainly describes the procedures at a BGP-LS Producer that originate link-state information into BGP-LS.

4.1. TLV Format

Information in the new Link-State NLRIs and the BGP-LS Attribute is encoded in Type/Length/Value triplets. The TLV format is shown in Figure 4 and applies to both the NLRI and the BGP-LS Attribute encodings.

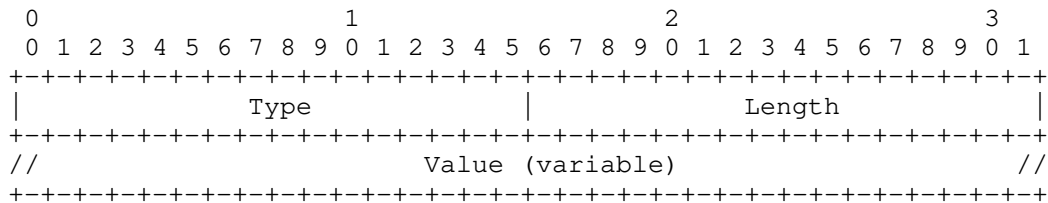


Figure 4: TLV Format

The Length field defines the length of the value portion in octets (thus, a TLV with no value portion would have a length of zero). The TLV is not padded to 4-octet alignment. Unknown and unsupported types MUST be preserved and propagated within both the NLRI and the BGP-LS Attribute. The presence of unrecognized or unexpected TLVs MUST NOT result in the NLRI or the BGP-LS Attribute being considered as malformed.

In order to compare NLRIs with unknown TLVs, all TLVs within the NLRI MUST be ordered in ascending order by TLV Type. If there are multiple TLVs of the same type within a single NLRI, then the TLVs sharing the same type MUST be in ascending order based on the value field. Comparison of the value fields is performed by treating the entire field as an opaque hexadecimal string. Standard string comparison rules apply. NLRIs having TLVs which do not follow the above ordering rules MUST be considered as malformed by a BGP-LS Propagator. This ensures that multiple copies of the same NLRI from multiple BGP-LS Producers and the ambiguity arising there from is prevented.

All TLVs within the NLRI that are not specified as mandatory are considered optional. All TLVs within the BGP-LS Attribute are considered optional unless specified otherwise.

The TLVs within the BGP-LS Attribute need not be ordered in any specific order.

4.2. The Link-State NLRI

The MP_REACH_NLRI and MP_UNREACH_NLRI attributes are BGP's containers for carrying opaque information. This specification defines three Link-State NLRI types that describes either a node, a link, and a prefix.

All non-VPN link, node, and prefix information SHALL be encoded using AFI 16388 / SAFI 71. VPN link, node, and prefix information SHALL be encoded using AFI 16388 / SAFI 72.

In order for two BGP speakers to exchange Link-State NLRI, they MUST use BGP Capabilities Advertisement to ensure that they are both capable of properly processing such NLRI. This is done as specified in [RFC4760], by using capability code 1 (multi-protocol BGP), with AFI 16388 / SAFI 71 for BGP-LS, and AFI 16388 / SAFI 72 for BGP-LS-VPN.

New Link-State NLRI Types may be introduced in the future. Since supported NLRI type values within the address family are not expressed in the Multiprotocol BGP (MP-BGP) capability [RFC4760], it is possible that a BGP speaker has advertised support for Link-State but does not support a particular Link-State NLRI type. In order to allow introduction of new Link-State NLRI types seamlessly in the future, without the need for upgrading all BGP speakers in the propagation path (e.g. a route reflector), this document deviates from the default handling behavior specified by [RFC7606] for Link-State address-family. An implementation MUST handle unrecognized Link-State NLRI types as opaque objects and MUST preserve and propagate them.

The format of the Link-State NLRI is shown in the following figures.

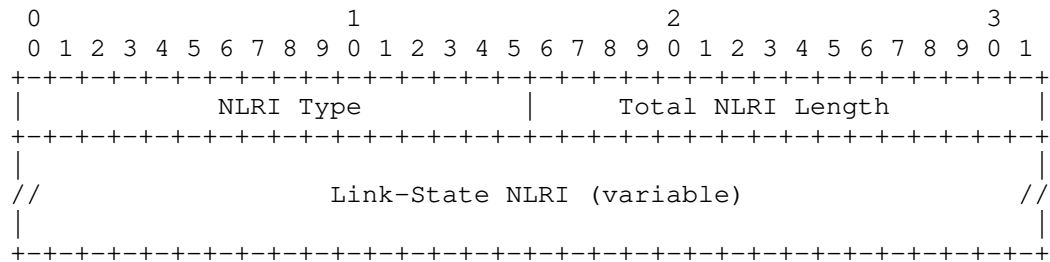


Figure 5: Link-State AFI 16388 / SAFI 71 NLRI Format

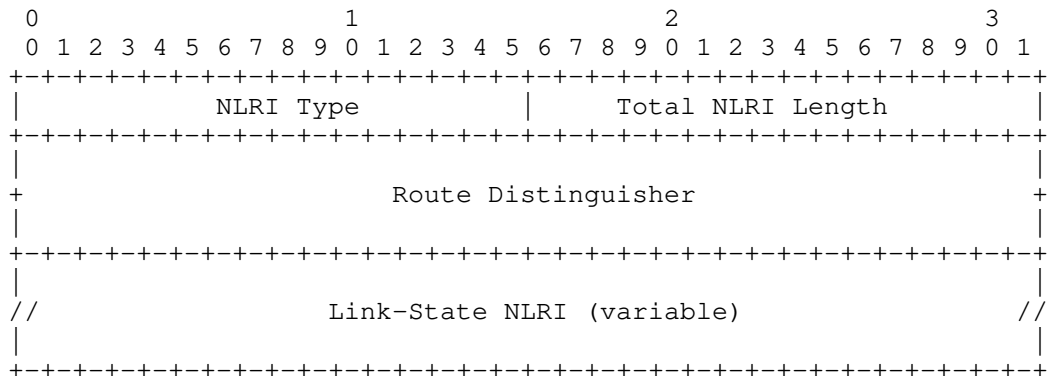


Figure 6: Link-State VPN AFI 16388 / SAFI 72 NLRI Format

The Total NLRI Length field contains the cumulative length, in octets, of the rest of the NLRI, not including the NLRI Type field or itself. For VPN applications, it also includes the length of the Route Distinguisher.

| Type | NLRI Type |
|-------------|---------------------------|
| 1 | Node NLRI |
| 2 | Link NLRI |
| 3 | IPv4 Topology Prefix NLRI |
| 4 | IPv6 Topology Prefix NLRI |
| 65000-65535 | Private Use |

Table 1: NLRI Types

Route Distinguishers are defined and discussed in [RFC4364].

The Node NLRI (NLRI Type = 1) is shown in the following figure.

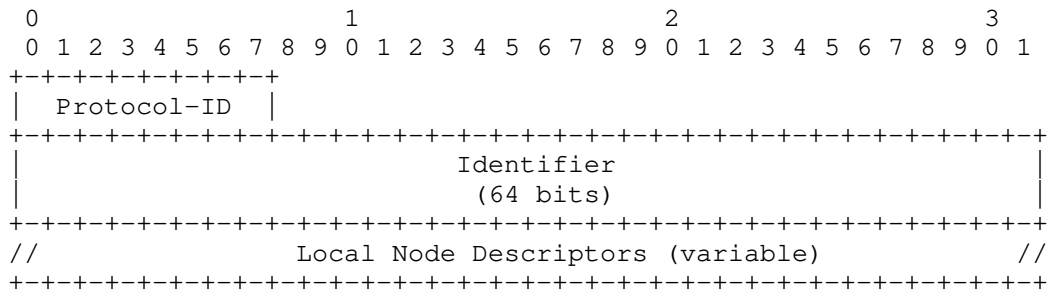


Figure 7: The Node NLRI Format

The Link NLRI (NLRI Type = 2) is shown in the following figure.

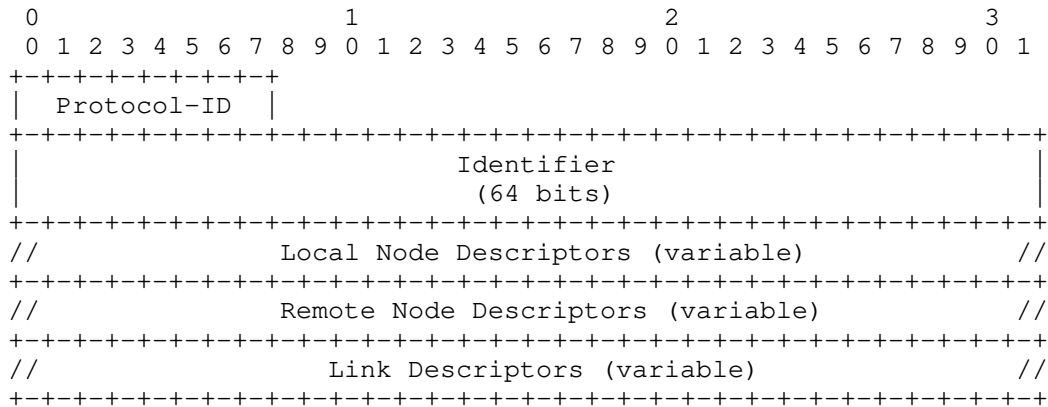


Figure 8: The Link NLRI Format

The IPv4 and IPv6 Prefix NLRIs (NLRI Type = 3 and Type = 4) use the same format, as shown in the following figure.

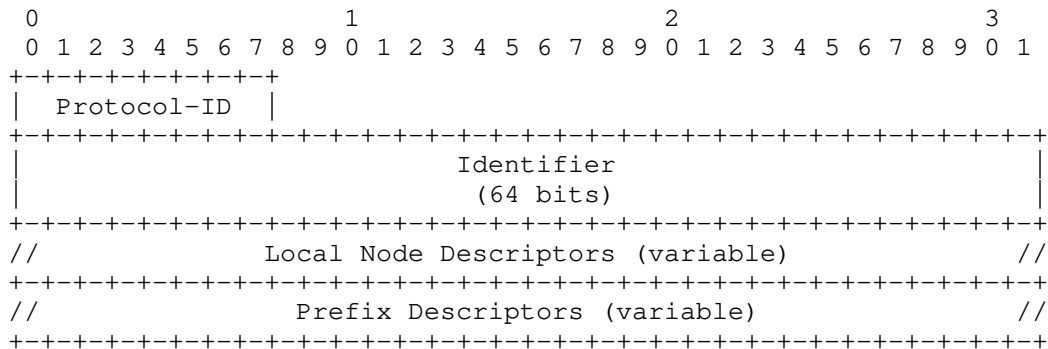


Figure 9: The IPv4/IPv6 Topology Prefix NLRI Format

The Protocol-ID field can contain one of the following values:

| Protocol-ID | NLRI information source protocol |
|-------------|----------------------------------|
| 1 | IS-IS Level 1 |
| 2 | IS-IS Level 2 |
| 3 | OSPFv2 |
| 4 | Direct |
| 5 | Static configuration |
| 6 | OSPFv3 |
| 200-255 | Private Use |

Table 2: Protocol Identifiers

The 'Direct' and 'Static configuration' protocol types SHOULD be used when BGP-LS is sourcing local information. For all information derived from other protocols, the corresponding Protocol-ID MUST be used. If BGP-LS has direct access to interface information and wants to advertise a local link, then the Protocol-ID 'Direct' SHOULD be used. For modeling virtual links, such as described in Section 5, the Protocol-ID 'Static configuration' SHOULD be used.

A router MAY run multiple protocol instances of OSPF or ISIS where by it becomes a border router between multiple IGP domains. Both OSPF and IS-IS MAY also run multiple routing protocol instances over the same link. See [RFC8202] and [RFC6549]. These instances define independent IGP routing domains. The 64-bit Identifier field carries a BGP-LS Instance Identifier (Instance-ID) that is used to identify the IGP routing domain where the NLRI belongs. The NLRIs representing link-state objects (nodes, links, or prefixes) from the same IGP routing instance MUST have the same Identifier field value.

NLRIs with different Identifier field values MUST be considered to be from different IGP routing instances. The Identifier field value 0 is RECOMMENDED to be used when there is only a single protocol instance in the network where BGP-LS is operational.

An implementation which supports multiple IGP instances MUST support the configuration of unique BGP-LS Instance-IDs at the routing protocol instance level. The network operator MUST assign consistent BGP-LS Instance-ID values on all BGP-LS Producers within a given IGP domain. Unique BGP-LS Instance-ID values MUST be assigned to routing protocol instances operating in different IGP domains. This allows the BGP-LS Consumer to build an accurate segregated multi-domain topology based on the Identifier field even when the topology is advertised via BGP-LS by multiple BGP-LS Producers in the network.

When the above described semantics and recommendations are not followed, a BGP-LS Consumer may see duplicate link-state objects for the same node, link or prefix when there are multiple BGP-LS Producers deployed. This may also result in the BGP-LS Consumers getting an inaccurate network-wide topology.

When adding, removing or modifying a TLV/sub-TLV from a Link-State NLRI, the BGP-LS Producer MUST withdraw the old NLRI by including it in the MP_UNREACH_NLRI. Not doing so can result in duplicate and inconsistent link-state objects hanging around in the BGP-LS table.

Each Node Descriptor and Link Descriptor consists of one or more TLVs, as described in the following sections.

4.2.1. Node Descriptors

Each link is anchored by a pair of Router-IDs that are used by the underlying IGP, namely, a 48-bit ISO System-ID for IS-IS and a 32-bit Router-ID for OSPFv2 and OSPFv3. An IGP may use one or more additional auxiliary Router-IDs, mainly for Traffic Engineering purposes. For example, IS-IS may have one or more IPv4 and IPv6 TE Router-IDs [RFC5305] [RFC6119]. These auxiliary Router-IDs MUST be included in the node attribute described in Section 4.3.1 and MAY be included in link attribute described in Section 4.3.2. The advertisement of the TE Router-IDs help a BGP-LS Consumer to correlate multiple link-state objects (e.g. in different IGP instances or areas/levels) to the same node in the network.

It is desirable that the Router-ID assignments inside the Node Descriptor are globally unique. However, there may be Router-ID spaces (e.g., ISO) where no global registry exists, or worse, Router-IDs have been allocated following the private-IP allocation described

in RFC 1918 [RFC1918]. BGP-LS uses the Autonomous System (AS) Number to disambiguate the Router-IDs, as described in Section 4.2.1.1.

4.2.1.1. Globally Unique Node/Link/Prefix Identifiers

One problem that needs to be addressed is the ability to identify an IGP node globally (by "globally", we mean within the BGP-LS database collected by all BGP-LS speakers that talk to each other). This can be expressed through the following two requirements:

- (A) The same node MUST NOT be represented by two keys (otherwise, one node will look like two nodes).
- (B) Two different nodes MUST NOT be represented by the same key (otherwise, two nodes will look like one node).

We define an "IGP domain" to be the set of nodes (hence, by extension links and prefixes) within which each node has a unique IGP representation by using the combination of Area-ID, Router-ID, Protocol-ID, Multi-Topology ID, and Instance-ID. The problem is that BGP may receive node/link/prefix information from multiple independent "IGP domains", and we need to distinguish between them. Moreover, we can't assume there is always one and only one IGP domain per AS. During IGP transitions, it may happen that two redundant IGPs are in place.

The mapping of the Instance-ID to the Identifier field as described earlier along with a set of sub-TLVs described in Section 4.2.1.4, allows specification of a flexible key for any given node/link information such that global uniqueness of the NLRI is ensured.

4.2.1.2. Local Node Descriptors

The Local Node Descriptors TLV contains Node Descriptors for the node anchoring the local end of the link. This is a mandatory TLV in all three types of NLRIs (node, link, and prefix). The Type is 256. The length of this TLV is variable. The value contains one or more Node Descriptor Sub-TLVs defined in Section 4.2.1.4.

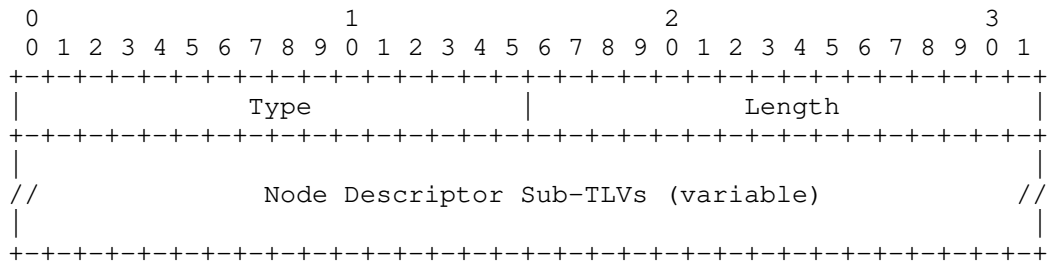


Figure 10: Local Node Descriptors TLV Format

4.2.1.3. Remote Node Descriptors

The Remote Node Descriptors TLV contains Node Descriptors for the node anchoring the remote end of the link. This is a mandatory TLV for Link NLRIs. The type is 257. The length of this TLV is variable. The value contains one or more Node Descriptor Sub-TLVs defined in Section 4.2.1.4.

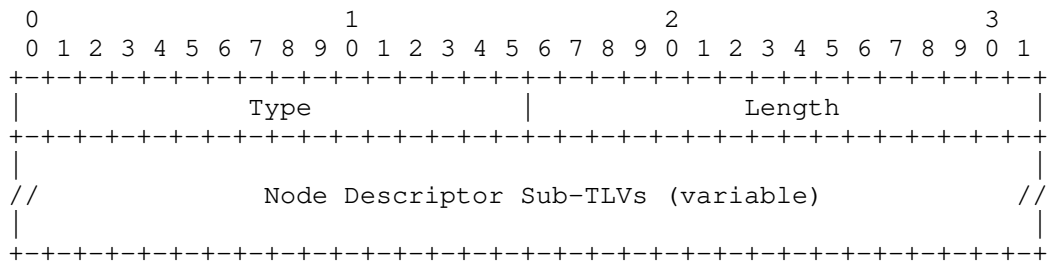


Figure 11: Remote Node Descriptors TLV Format

4.2.1.4. Node Descriptor Sub-TLVs

The Node Descriptor Sub-TLV type code points and lengths are listed in the following table:

| Sub-TLV Code Point | Description | Length |
|--------------------|--------------------------------|----------|
| 512 | Autonomous System | 4 |
| 513 | BGP-LS Identifier (deprecated) | 4 |
| 514 | OSPF Area-ID | 4 |
| 515 | IGP Router-ID | Variable |

Table 3: Node Descriptor Sub-TLVs

The sub-TLV values in Node Descriptor TLVs are defined as follows:

Autonomous System: Opaque value (32-bit AS Number). This is an optional TLV. The value SHOULD be set to the AS Number associated with the BGP process originating the link-state information. An implementation MAY provide a configuration option on the BGP-LS Producer to use a value different.

BGP-LS Identifier: Opaque value (32-bit ID). This is an optional TLV. In conjunction with Autonomous System Number (ASN), uniquely identifies the BGP-LS domain. The combination of ASN and BGP-LS ID MUST be globally unique. All BGP-LS speakers within an IGP flooding-set (set of IGP nodes within which an LSP/LSA is flooded) MUST use the same ASN, BGP-LS ID tuple. If an IGP domain consists of multiple flooding-sets, then all BGP-LS speakers within the IGP domain SHOULD use the same ASN, BGP-LS ID tuple.

Area-ID: Used to identify the 32-bit area to which the NLRI belongs. This is a mandatory TLV when originating information from OSPF. The Area Identifier allows different NLRIs of the same router to be discriminated.

IGP Router-ID: Opaque value. This is a mandatory TLV when originating information from IS-IS, OSPF, direct or static. For an IS-IS non-pseudonode, this contains a 6-octet ISO Node-ID (ISO system-ID). For an IS-IS pseudonode corresponding to a LAN, this contains the 6-octet ISO Node-ID of the Designated Intermediate System (DIS) followed by a 1-octet, nonzero PSN identifier (7 octets in total). For an OSPFv2 or OSPFv3 non-pseudonode, this contains the 4-octet Router-ID. For an OSPFv2 pseudonode representing a LAN, this contains the 4-octet Router-ID of the Designated Router (DR) followed by the 4-octet IPv4 address of the DR's interface to the LAN (8 octets in total). Similarly, for an OSPFv3 pseudonode, this contains the 4-octet Router-ID of the DR followed by the 4-octet interface identifier of the DR's interface to the LAN (8 octets in total). The TLV size in combination with the protocol identifier enables the decoder to determine the type of the node. For Direct or Static configuration, the value SHOULD be taken from an IPv4 or IPv6 address (e.g. loopback interface) configured on the node.

There can be at most one instance of each sub-TLV type present in any Node Descriptor. The sub-TLVs within a Node Descriptor MUST be arranged in ascending order by sub-TLV type. This needs to be done in order to compare NLRIs, even when an implementation encounters an unknown sub-TLV. Using stable sorting, an implementation can do binary comparison of NLRIs and hence allow incremental deployment of new key sub-TLVs.

The BGP-LS Identifier was introduced by [RFC7752] and it's use is being deprecated by this document. Implementations MUST continue to support this sub-TLV for backward compatibility. The default value of 0 is RECOMMENDED to be use when a BGP-LS Producer includes this sub-TLV when originating information into BGP-LS. Implementations MAY provide an option to configure this value for backward compatibility reasons. The use of the Instance-ID in the Identifier field is the RECOMMENDED way of segregation of different IGP domains in BGP-LS.

4.2.2. Link Descriptors

The Link Descriptor field is a set of Type/Length/Value (TLV) triplets. The format of each TLV is shown in Section 4.1. The Link Descriptor TLVs uniquely identify a link among multiple parallel links between a pair of anchor routers. A link described by the Link Descriptor TLVs actually is a "half-link", a unidirectional representation of a logical link. In order to fully describe a single logical link, two originating routers advertise a half-link each, i.e., two Link NLRIs are advertised for a given point-to-point link.

A BGP-LS Consumer should not consider a link between two nodes as being available unless it has received the two Link NLRIs corresponding to the half-link representation of that link from both the nodes. This check is similar to the 'two way connectivity check' that is performed by link-state IGPs and is also required to be done by BGP-LS Consumers of link-state topology.

A BGP-LS Producer MAY suppress the advertisement of a Link NLRI, corresponding to a half link, from a link-state IGP unless it has verified that the link is being reported in the IS-IS LSP or OSPF Router LSA by both the nodes connected by that link. This 'two way connectivity check' is performed by link-state IGPs during their computation and may be leveraged before passing information for any half-link that is reported from these IGPs in to BGP-LS. This ensures that only those Link State IGP adjacencies which are established get reported via Link NLRIs. Such a 'two way connectivity check' may be also required in certain cases (e.g. with OSPF) to obtain the proper link identifiers of the remote node.

The format and semantics of the Value fields in most Link Descriptor TLVs correspond to the format and semantics of Value fields in IS-IS Extended IS Reachability sub-TLVs, defined in [RFC5305], [RFC5307], and [RFC6119]. Although the encodings for Link Descriptor TLVs were originally defined for IS-IS, the TLVs can carry data sourced by either IS-IS or OSPF.

The following TLVs are defined as Link Descriptors in the Link NLRI:

| TLV Code Point | Description | IS-IS TLV /Sub-TLV | Reference (RFC/Section) |
|----------------|-------------------------------|--------------------|-------------------------|
| 258 | Link Local/Remote Identifiers | 22/4 | [RFC5307]/1.1 |
| 259 | IPv4 interface address | 22/6 | [RFC5305]/3.2 |
| 260 | IPv4 neighbor address | 22/8 | [RFC5305]/3.3 |
| 261 | IPv6 interface address | 22/12 | [RFC6119]/4.2 |
| 262 | IPv6 neighbor address | 22/13 | [RFC6119]/4.3 |
| 263 | Multi-Topology Identifier | --- | Section 4.2.2.1 |

Table 4: Link Descriptor TLVs

The information about a link present in the LSA/LSP originated by the local node of the link determines the set of TLVs in the Link Descriptor of the link.

If interface and neighbor addresses, either IPv4 or IPv6, are present, then the IP address TLVs MUST be included and the Link Local/Remote Identifiers TLV MUST NOT be included in the Link Descriptor. The Link Local/Remote Identifiers TLV MAY be included in the link attribute when available.

If interface and neighbor addresses are not present and the link local/remote identifiers are present, then the Link Local/Remote Identifiers TLV MUST be included in the Link Descriptor.

The Multi-Topology Identifier TLV MUST be included in Link Descriptor if the underlying IGP link object is associated with a non-default topology.

4.2.2.1. Multi-Topology ID

The Multi-Topology ID (MT-ID) TLV carries one or more IS-IS or OSPF Multi-Topology IDs for a link, node, or prefix.

Semantics of the IS-IS MT-ID are defined in Section 7.2 of RFC 5120 [RFC5120]. Semantics of the OSPF MT-ID are defined in Section 3.7 of RFC 4915 [RFC4915]. Bits R are reserved and SHOULD be set to 0 when

originated and ignored on receipt. If the value in the MT-ID TLV is derived from OSPF, then the upper 5 bits of the MT-ID field MUST be set to 0.

The format of the MT-ID TLV is shown in the following figure.

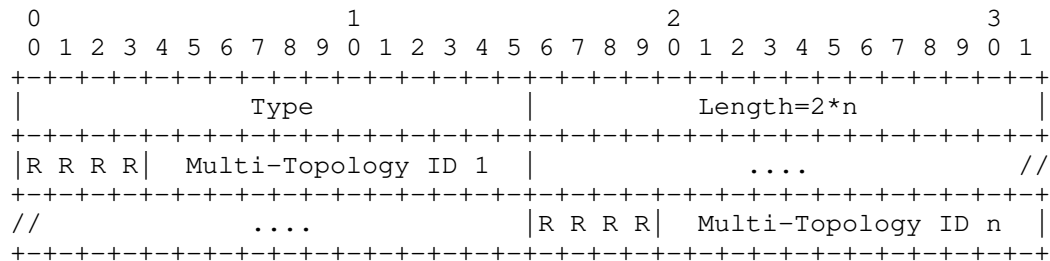


Figure 12: Multi-Topology ID TLV Format

where Type is 263, Length is $2*n$, and n is the number of MT-IDs carried in the TLV.

The MT-ID TLV MAY be present in a Link Descriptor, a Prefix Descriptor, or the BGP-LS attribute of a Node NLRI. In a Link or Prefix Descriptor, only a single MT-ID TLV containing the MT-ID of the topology where the link or the prefix is reachable is allowed. In case one wants to advertise multiple topologies for a given Link Descriptor or Prefix Descriptor, multiple NLRIs MUST be generated where each NLRI contains a single unique MT-ID. In the BGP-LS attribute of a Node NLRI, one MT-ID TLV containing the array of MT-IDs of all topologies where the node is reachable is allowed.

4.2.3. Prefix Descriptors

The Prefix Descriptor field is a set of Type/Length/Value (TLV) triplets. Prefix Descriptor TLVs uniquely identify an IPv4 or IPv6 prefix originated by a node. The following TLVs are defined as Prefix Descriptors in the IPv4/IPv6 Prefix NLRI:

| TLV Code Point | Description | Length | Reference (RFC/Section) |
|----------------|-----------------------------|----------|-------------------------|
| 263 | Multi-Topology Identifier | variable | Section 4.2.2.1 |
| 264 | OSPF Route Type | 1 | Section 4.2.3.1 |
| 265 | IP Reachability Information | variable | Section 4.2.3.2 |

Table 5: Prefix Descriptor TLVs

The Multi-Topology Identifier TLV MUST be included in Prefix Descriptor if the underlying IGP prefix object is associated with a non-default topology.

4.2.3.1. OSPF Route Type

The OSPF Route Type TLV is a mandatory TLV corresponding to Prefix NLRIs originated from OSPF. It is used to identify the OSPF route type of the prefix. An OSPF prefix MAY be advertised in the OSPF domain with multiple route types. The Route Type TLV allows the discrimination of these advertisements. The format of the OSPF Route Type TLV is shown in the following figure.

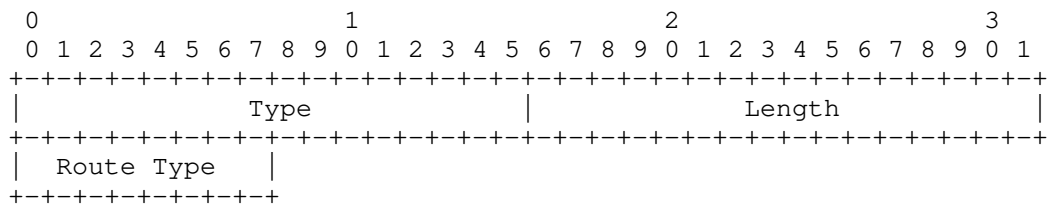


Figure 13: OSPF Route Type TLV Format

where the Type and Length fields of the TLV are defined in Table 5. The OSPF Route Type field values are defined in the OSPF protocol and can be one of the following:

- o Intra-Area (0x1)
- o Inter-Area (0x2)
- o External 1 (0x3)
- o External 2 (0x4)

- o NSSA 1 (0x5)
- o NSSA 2 (0x6)

4.2.3.2. IP Reachability Information

The IP Reachability Information TLV is a mandatory TLV for IPv4 & IPv6 Prefix NLRI types. The TLV contains one IP address prefix (IPv4 or IPv6) originally advertised in the IGP topology. Its purpose is to glue a particular BGP service NLRI by virtue of its BGP next hop to a given node in the LSDB. A router SHOULD advertise an IP Prefix NLRI for each of its BGP next hops. The format of the IP Reachability Information TLV is shown in the following figure:

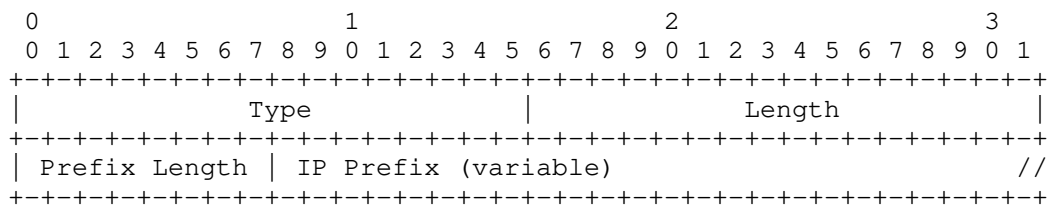


Figure 14: IP Reachability Information TLV Format

The Type and Length fields of the TLV are defined in Table 5. The following two fields determine the reachability information of the address family. The Prefix Length field contains the length of the prefix in bits. The IP Prefix field contains the most significant octets of the prefix, i.e., 1 octet for prefix length 1 up to 8, 2 octets for prefix length 9 to 16, 3 octets for prefix length 17 up to 24, 4 octets for prefix length 25 up to 32, etc.

4.3. The BGP-LS Attribute

The BGP-LS Attribute is an optional, non-transitive BGP attribute that is used to carry link, node, and prefix parameters and attributes. It is defined as a set of Type/Length/Value (TLV) triplets, described in the following section. This attribute SHOULD only be included with Link-State NLRIs. This attribute MUST be ignored for all other address families.

The Node Attribute TLVs, Link Attribute TLVs and Prefix Attribute TLVs are sets of TLVs that may be encoded in the BGP-LS Attribute associated with a Node NLRI, Link NLRI and Prefix NLRI respectively.

The BGP-LS Attribute may potentially grow large in size depending on the amount of link-state information associated with a single Link-State NLRI. The BGP specification [RFC4271] mandates a maximum BGP

message size of 4096 octets. It is RECOMMENDED that an implementation support [I-D.ietf-idr-bgp-extended-messages] in order to accommodate larger size of information within the BGP-LS Attribute. BGP-LS Producers MUST ensure that they limit the TLVs included in the BGP-LS Attribute to ensure that a BGP update message for a single Link-State NLRI does not cross the maximum limit for a BGP message. The determination of the types of TLVs to be included MAY be made by the BGP-LS Producer based on the BGP-LS Consumer applications requirement and is outside the scope of this document. When a BGP-LS Propagator finds that it is exceeding the maximum BGP message size due to addition or update of some other BGP Attribute (e.g. AS_PATH), it MUST consider the BGP-LS Attribute to be malformed and handle the propagation as described in Section 7.2.2.

4.3.1. Node Attribute TLVs

The following Node Attribute TLVs are defined for the BGP-LS Attribute associated with a Node NLRI:

| TLV Code Point | Description | Length | Reference (RFC/Section) |
|----------------|------------------------------|----------|-------------------------|
| 263 | Multi-Topology Identifier | variable | Section 4.2.2.1 |
| 1024 | Node Flag Bits | 1 | Section 4.3.1.1 |
| 1025 | Opaque Node Attribute | variable | Section 4.3.1.5 |
| 1026 | Node Name | variable | Section 4.3.1.3 |
| 1027 | IS-IS Area Identifier | variable | Section 4.3.1.2 |
| 1028 | IPv4 Router-ID of Local Node | 4 | [RFC5305]/4.3 |
| 1029 | IPv6 Router-ID of Local Node | 16 | [RFC6119]/4.1 |

Table 6: Node Attribute TLVs

4.3.1.1. Node Flag Bits TLV

The Node Flag Bits TLV carries a bit mask describing node attributes. The value is a variable-length bit array of flags, where each bit represents a node capability.

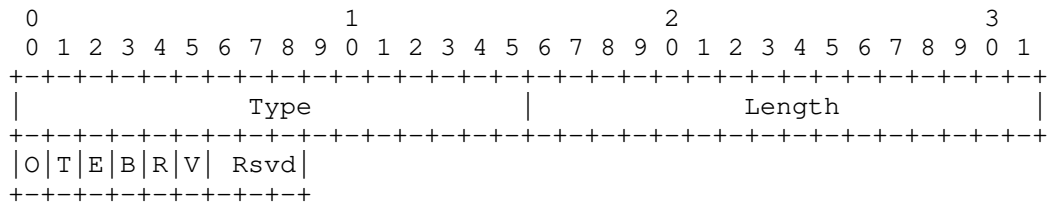


Figure 15: Node Flag Bits TLV Format

The bits are defined as follows:

| Bit | Description | Reference |
|-----------------|-------------------------|------------|
| 'O' | Overload Bit | [ISO10589] |
| 'T' | Attached Bit | [ISO10589] |
| 'E' | External Bit | [RFC2328] |
| 'B' | ABR Bit | [RFC2328] |
| 'R' | Router Bit | [RFC5340] |
| 'V' | V6 Bit | [RFC5340] |
| Reserved (Rsvd) | Reserved for future use | |

Table 7: Node Flag Bits Definitions

4.3.1.2. IS-IS Area Identifier TLV

An IS-IS node can be part of one or more IS-IS areas. Each of these area addresses is carried in the IS-IS Area Identifier TLV. If multiple area addresses are present, multiple TLVs are used to encode them. The IS-IS Area Identifier TLV may be present in the BGP-LS attribute only when advertised in the Link-State Node NLRI.

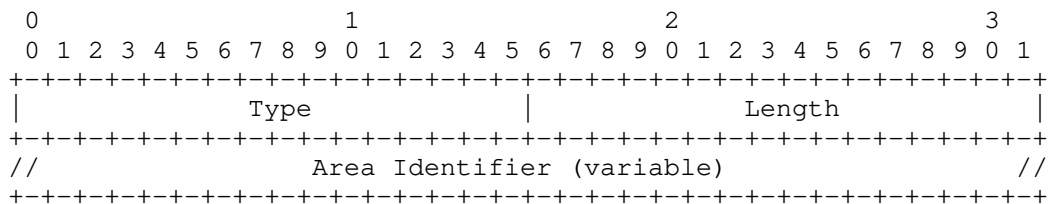


Figure 16: IS-IS Area Identifier TLV Format

4.3.1.3. Node Name TLV

The Node Name TLV is optional. Its structure and encoding has been borrowed from [RFC5301]. The Value field identifies the symbolic name of the router node. This symbolic name can be the Fully Qualified Domain Name (FQDN) for the router, it can be a subset of the FQDN (e.g., a hostname), or it can be any string operators want to use for the router. The use of FQDN or a subset of it is strongly RECOMMENDED. The maximum length of the Node Name TLV is 255 octets.

The Value field is encoded in 7-bit ASCII. If a user interface for configuring or displaying this field permits Unicode characters, that user interface is responsible for applying the ToASCII and/or ToUnicode algorithm as described in [RFC5890] to achieve the correct format for transmission or display.

[RFC5301] describes an IS-IS-specific extension and [RFC5642] describes an OSPF extension for advertisement of Node Name which MAY encoded in the Node Name TLV.

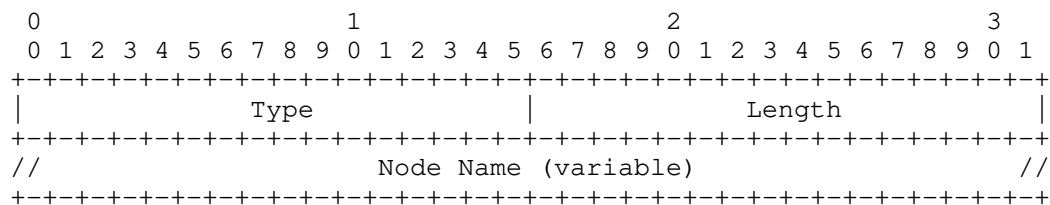


Figure 17: Node Name Format

4.3.1.4. Local IPv4/IPv6 Router-ID TLVs

The local IPv4/IPv6 Router-ID TLVs are used to describe auxiliary Router-IDs that the IGP might be using, e.g., for TE and migration purposes such as correlating a Node-ID between different protocols. If there is more than one auxiliary Router-ID of a given type, then each one is encoded in its own TLV.

4.3.1.5. Opaque Node Attribute TLV

The Opaque Node Attribute TLV is an envelope that transparently carries optional Node Attribute TLVs advertised by a router. An originating router shall use this TLV for encoding information specific to the protocol advertised in the NLRI header Protocol-ID field or new protocol extensions to the protocol as advertised in the NLRI header Protocol-ID field for which there is no protocol-neutral representation in the BGP Link-State NLRI. The primary use of the Opaque Node Attribute TLV is to bridge the document lag between,

e.g., a new IGP link-state attribute being defined and the protocol-neutral BGP-LS extensions being published. A router, for example, could use this extension in order to advertise the native protocol's Node Attribute TLVs, such as the OSPF Router Informational Capabilities TLV defined in [RFC7770] or the IGP TE Node Capability Descriptor TLV described in [RFC5073].

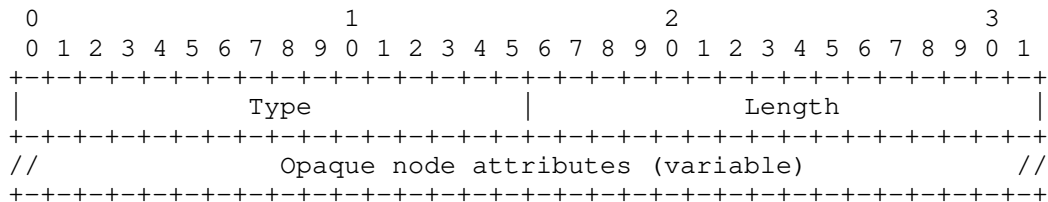


Figure 18: Opaque Node Attribute Format

4.3.2. Link Attribute TLVs

Link Attribute TLVs are TLVs that may be encoded in the BGP-LS attribute with a Link NLRI. Each 'Link Attribute' is a Type/Length/Value (TLV) triplet formatted as defined in Section 4.1. The format and semantics of the Value fields in some Link Attribute TLVs correspond to the format and semantics of the Value fields in IS-IS Extended IS Reachability sub-TLVs, defined in [RFC5305] and [RFC5307]. Other Link Attribute TLVs are defined in this document. Although the encodings for Link Attribute TLVs were originally defined for IS-IS, the TLVs can carry data sourced by either IS-IS or OSPF.

The following Link Attribute TLVs are defined for the BGP-LS Attribute associated with a Link NLRI:

| TLV Code Point | Description | IS-IS TLV /Sub-TLV | Reference (RFC/Section) |
|----------------|--------------------------------|--------------------|-------------------------|
| 1028 | IPv4 Router-ID of Local Node | 134/--- | [RFC5305]/4.3 |
| 1029 | IPv6 Router-ID of Local Node | 140/--- | [RFC6119]/4.1 |
| 1030 | IPv4 Router-ID of Remote Node | 134/--- | [RFC5305]/4.3 |
| 1031 | IPv6 Router-ID of Remote Node | 140/--- | [RFC6119]/4.1 |
| 1088 | Administrative group (color) | 22/3 | [RFC5305]/3.1 |
| 1089 | Maximum link bandwidth | 22/9 | [RFC5305]/3.4 |
| 1090 | Max. reservable link bandwidth | 22/10 | [RFC5305]/3.5 |
| 1091 | Unreserved bandwidth | 22/11 | [RFC5305]/3.6 |
| 1092 | TE Default Metric | 22/18 | Section 4.3.2.3 |
| 1093 | Link Protection Type | 22/20 | [RFC5307]/1.2 |
| 1094 | MPLS Protocol Mask | --- | Section 4.3.2.2 |
| 1095 | IGP Metric | --- | Section 4.3.2.4 |
| 1096 | Shared Risk Link Group | --- | Section 4.3.2.5 |
| 1097 | Opaque Link Attribute | --- | Section 4.3.2.6 |
| 1098 | Link Name | --- | Section 4.3.2.7 |

Table 8: Link Attribute TLVs

4.3.2.1. IPv4/IPv6 Router-ID TLVs

The local/remote IPv4/IPv6 Router-ID TLVs are used to describe auxiliary Router-IDs that the IGP might be using, e.g., for TE purposes. All auxiliary Router-IDs of both the local and the remote node MUST be included in the link attribute of each Link NLRI. If there is more than one auxiliary Router-ID of a given type, then multiple TLVs are used to encode them.

4.3.2.2. MPLS Protocol Mask TLV

The MPLS Protocol Mask TLV carries a bit mask describing which MPLS signaling protocols are enabled. The length of this TLV is 1. The

value is a bit array of 8 flags, where each bit represents an MPLS Protocol capability.

Generation of the MPLS Protocol Mask TLV is only valid for and SHOULD only be used with originators that have local link insight, for example, the Protocol-IDs 'Static configuration' or 'Direct' as per Table 2. The MPLS Protocol Mask TLV MUST NOT be included in NLRIs with the other Protocol-IDs listed in Table 2.

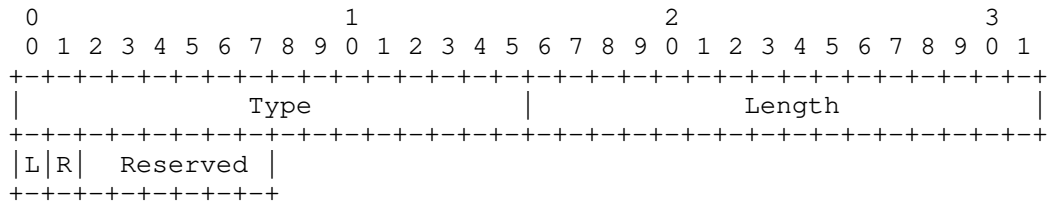


Figure 19: MPLS Protocol Mask TLV

The following bits are defined:

| Bit | Description | Reference |
|------------|---|-----------|
| 'L' | Label Distribution Protocol (LDP) | [RFC5036] |
| 'R' | Extension to RSVP for LSP Tunnels (RSVP-TE) | [RFC3209] |
| 'Reserved' | Reserved for future use | |

Table 9: MPLS Protocol Mask TLV Codes

4.3.2.3. TE Default Metric TLV

The TE Default Metric TLV carries the Traffic Engineering metric for this link. The length of this TLV is fixed at 4 octets. If a source protocol uses a metric width of less than 32 bits, then the high-order bits of this field MUST be padded with zero.

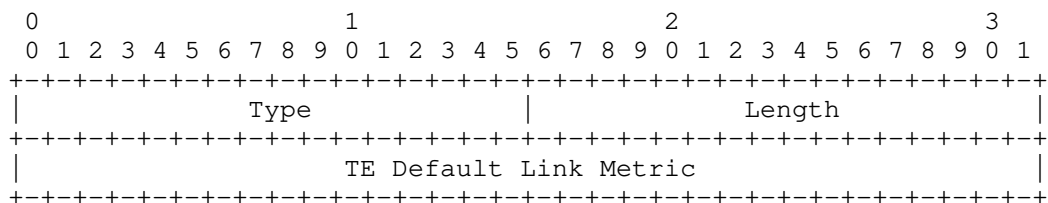


Figure 20: TE Default Metric TLV Format

4.3.2.4. IGP Metric TLV

The IGP Metric TLV carries the metric for this link. The length of this TLV is variable, depending on the metric width of the underlying protocol. IS-IS small metrics have a length of 1 octet (the two most significant bits are ignored). OSPF link metrics have a length of 2 octets. IS-IS wide metrics have a length of 3 octets.

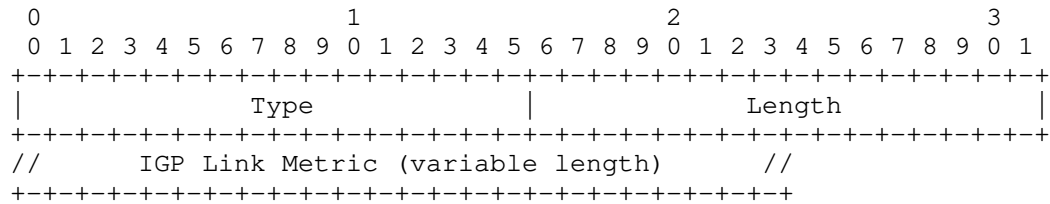


Figure 21: IGP Metric TLV Format

4.3.2.5. Shared Risk Link Group TLV

The Shared Risk Link Group (SRLG) TLV carries the Shared Risk Link Group information (see Section 2.3 ("Shared Risk Link Group Information") of [RFC4202]). It contains a data structure consisting of a (variable) list of SRLG values, where each element in the list has 4 octets, as shown in Figure 22. The length of this TLV is 4 * (number of SRLG values).

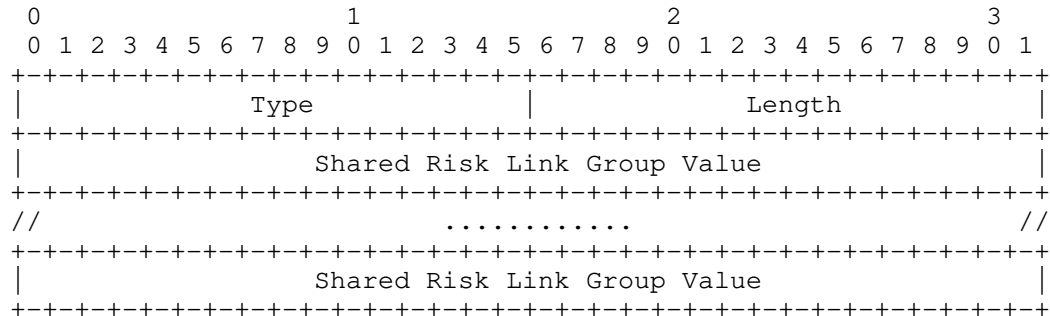


Figure 22: Shared Risk Link Group TLV Format

The SRLG TLV for OSPF-TE is defined in [RFC4203]. In IS-IS, the SRLG information is carried in two different TLVs: the IPv4 (SRLG) TLV (Type 138) defined in [RFC5307] and the IPv6 SRLG TLV (Type 139) defined in [RFC6119]. In Link-State NLRI, both IPv4 and IPv6 SRLG information are carried in a single TLV.

4.3.2.6. Opaque Link Attribute TLV

The Opaque Link Attribute TLV is an envelope that transparently carries optional Link Attribute TLVs advertised by a router. An originating router shall use this TLV for encoding information specific to the protocol advertised in the NLRI header Protocol-ID field or new protocol extensions to the protocol as advertised in the NLRI header Protocol-ID field for which there is no protocol-neutral representation in the BGP Link-State NLRI. The primary use of the Opaque Link Attribute TLV is to bridge the document lag between, e.g., a new IGP link-state attribute being defined and the 'protocol-neutral' BGP-LS extensions being published.

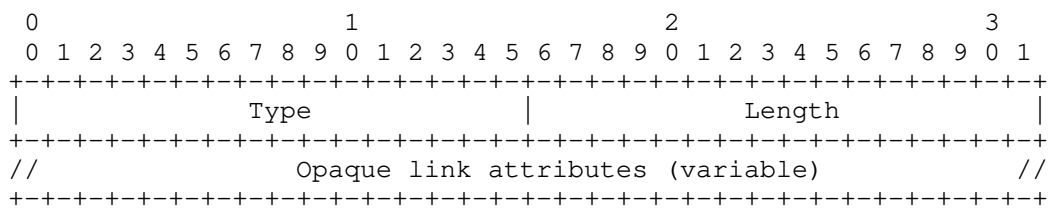


Figure 23: Opaque Link Attribute TLV Format

4.3.2.7. Link Name TLV

The Link Name TLV is optional. The Value field identifies the symbolic name of the router link. This symbolic name can be the FQDN for the link, it can be a subset of the FQDN, or it can be any string operators want to use for the link. The use of FQDN or a subset of it is strongly RECOMMENDED. The maximum length of the Link Name TLV is 255 octets.

The Value field is encoded in 7-bit ASCII. If a user interface for configuring or displaying this field permits Unicode characters, that user interface is responsible for applying the ToASCII and/or ToUnicode algorithm as described in [RFC5890] to achieve the correct format for transmission or display.

How a router derives and injects link names is outside of the scope of this document.

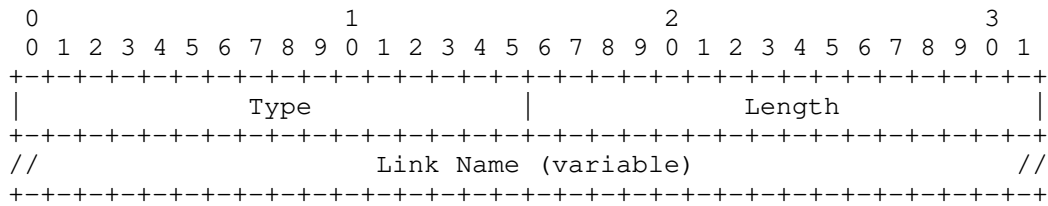


Figure 24: Link Name TLV Format

4.3.3. Prefix Attribute TLVs

Prefixes are learned from the IGP topology (IS-IS or OSPF) with a set of IGP attributes (such as metric, route tags, etc.) that are advertised in the BGP-LS Attribute with Prefix NLRI types 3 and 4.

The following Prefix Attribute TLVs are defined for the BGP-LS Attribute associated with a Prefix NLRI:

| TLV Code Point | Description | Length | Reference |
|----------------|-------------------------|----------|-----------------|
| 1152 | IGP Flags | 1 | Section 4.3.3.1 |
| 1153 | IGP Route Tag | 4*n | [RFC5130] |
| 1154 | IGP Extended Route Tag | 8*n | [RFC5130] |
| 1155 | Prefix Metric | 4 | [RFC5305] |
| 1156 | OSPF Forwarding Address | 4 | [RFC2328] |
| 1157 | Opaque Prefix Attribute | variable | Section 4.3.3.6 |

Table 10: Prefix Attribute TLVs

4.3.3.1. IGP Flags TLV

The IGP Flags TLV contains IS-IS and OSPF flags and bits originally assigned to the prefix. The IGP Flags TLV is encoded as follows:

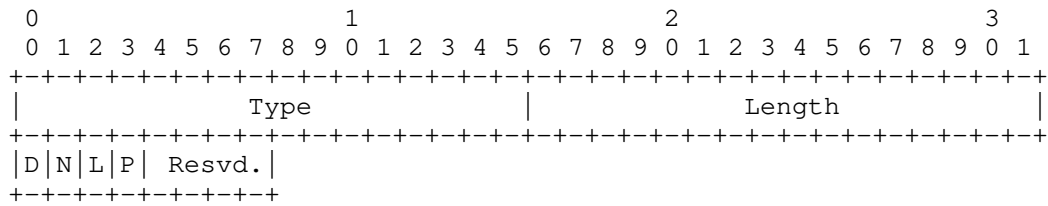


Figure 25: IGP Flag TLV Format

The Value field contains bits defined according to the table below:

| Bit | Description | Reference |
|----------|---------------------------|-----------|
| 'D' | IS-IS Up/Down Bit | [RFC5305] |
| 'N' | OSPF "no unicast" Bit | [RFC5340] |
| 'L' | OSPF "local address" Bit | [RFC5340] |
| 'P' | OSPF "propagate NSSA" Bit | [RFC5340] |
| Reserved | Reserved for future use. | |

Table 11: IGP Flag Bits Definitions

4.3.3.2. IGP Route Tag TLV

The IGP Route Tag TLV carries original IGP Tags (IS-IS [RFC5130] or OSPF) of the prefix and is encoded as follows:

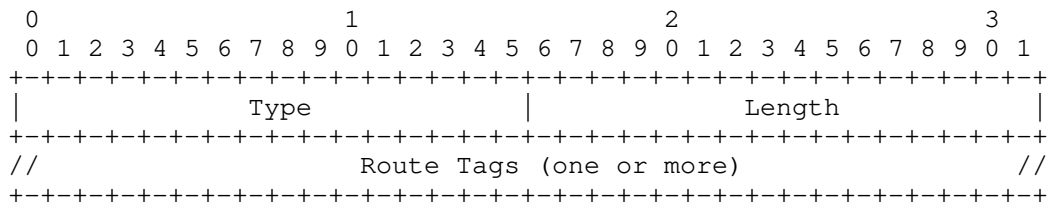


Figure 26: IGP Route Tag TLV Format

Length is a multiple of 4.

The Value field contains one or more Route Tags as learned in the IGP topology.

4.3.3.3. Extended IGP Route Tag TLV

The Extended IGP Route Tag TLV carries IS-IS Extended Route Tags of the prefix [RFC5130] and is encoded as follows:

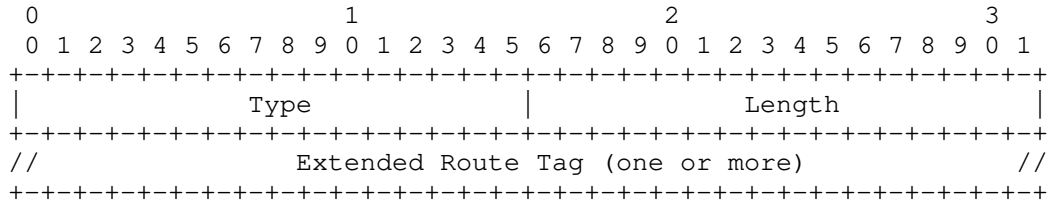


Figure 27: Extended IGP Route Tag TLV Format

Length is a multiple of 8.

The Extended Route Tag field contains one or more Extended Route Tags as learned in the IGP topology.

4.3.3.4. Prefix Metric TLV

The Prefix Metric TLV is an optional attribute and may only appear once. If present, it carries the metric of the prefix as known in the IGP topology as described in Section 4 of [RFC5305] (and therefore represents the reachability cost to the prefix). If not present, it means that the prefix is advertised without any reachability.

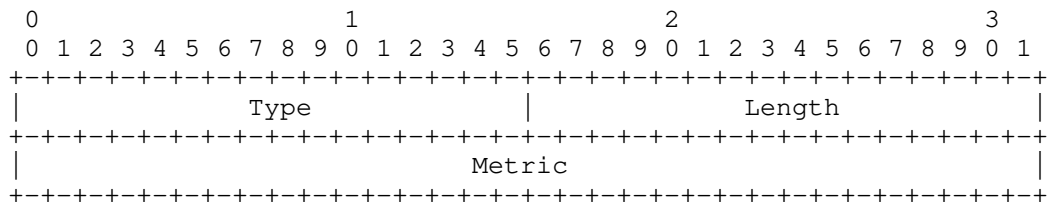


Figure 28: Prefix Metric TLV Format

Length is 4.

4.3.3.5. OSPF Forwarding Address TLV

The OSPF Forwarding Address TLV [RFC2328] [RFC5340] carries the OSPF forwarding address as known in the original OSPF advertisement. Forwarding address can be either IPv4 or IPv6.

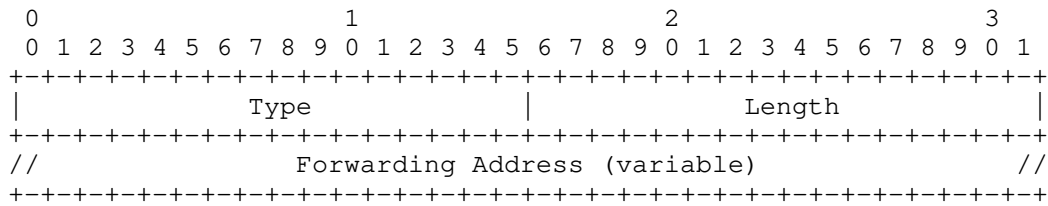


Figure 29: OSPF Forwarding Address TLV Format

Length is 4 for an IPv4 forwarding address, and 16 for an IPv6 forwarding address.

4.3.3.6. Opaque Prefix Attribute TLV

The Opaque Prefix Attribute TLV is an envelope that transparently carries optional Prefix Attribute TLVs advertised by a router. An originating router shall use this TLV for encoding information specific to the protocol advertised in the NLRI header Protocol-ID field or new protocol extensions to the protocol as advertised in the NLRI header Protocol-ID field for which there is no protocol-neutral representation in the BGP Link-State NLRI. The primary use of the Opaque Prefix Attribute TLV is to bridge the document lag between, e.g., a new IGP link-state attribute being defined and the protocol-neutral BGP-LS extensions being published.

The format of the TLV is as follows:

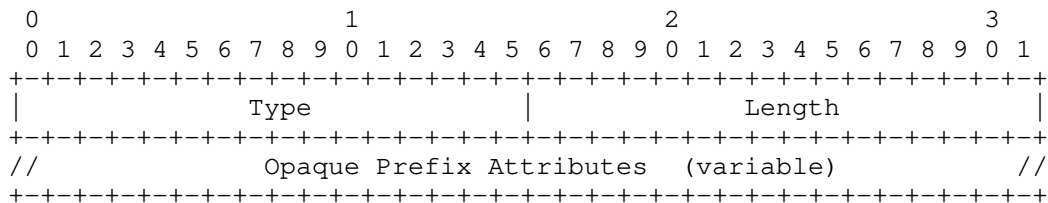


Figure 30: Opaque Prefix Attribute TLV Format

Type is as specified in Table 10. Length is variable.

4.4. Private Use

TLVs for Vendor Private use are supported using the code point range reserved as indicated in Section 6. For such TLV use in the NLRI or BGP-LS Attribute, the format as described in Section 4.1 is to be used and a 4 octet field MUST be included as the first field in the value to carry the Enterprise Code. For a private use NLRI Type, a 4 octet field MUST be included as the first field in the NLRI

immediately following the Total NLRI Length field of the Link-State NLRI format as described in Section 4.2 to carry the Enterprise Code. The Enterprise Codes are listed at <http://www.iana.org/assignments/enterprise-numbers>. This enables use vendor specific extensions without conflicts.

4.5. BGP Next-Hop Information

BGP link-state information for both IPv4 and IPv6 networks can be carried over either an IPv4 BGP session or an IPv6 BGP session. If an IPv4 BGP session is used, then the next hop in the MP_REACH_NLRI SHOULD be an IPv4 address. Similarly, if an IPv6 BGP session is used, then the next hop in the MP_REACH_NLRI SHOULD be an IPv6 address. Usually, the next hop will be set to the local endpoint address of the BGP session. The next-hop address MUST be encoded as described in [RFC4760]. The Length field of the next-hop address will specify 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; and if the next-hop length is 32, then there is one global IPv6 address followed by a link-local IPv6 address. The link-local IPv6 address should be used as described in [RFC2545]. For VPN Subsequent Address Family Identifier (SAFI), as per custom, an 8-byte Route Distinguisher set to all zero is prepended to the next hop.

The BGP Next Hop attribute is used by each BGP-LS speaker to validate the NLRI it receives. In case identical NLRIs are sourced by multiple BGP-LS Producers, the BGP Next Hop attribute is used to tiebreak as per the standard BGP path decision process. This specification doesn't mandate any rule regarding the rewrite of the BGP Next Hop attribute.

4.6. Inter-AS Links

The main source of TE information is the IGP, which is not active on inter-AS links. In some cases, the IGP may have information of inter-AS links [RFC5392] [RFC5316]. In other cases, an implementation SHOULD provide a means to inject inter-AS links into BGP-LS. The exact mechanism used to provision the inter-AS links is outside the scope of this document

4.7. Handling of Unreachable IGP Nodes

The origination and propagation of IGP link-state information via BGP needs to provide a consistent and true view of the topology of the IGP domain. BGP-LS provides an abstraction of the protocol specifics and BGP-LS Consumers may be varied types of applications.

A BGP-LS Consumer talks to a BGP route-reflector (RR) R0 which is aggregating the BGP-LS feed from the BGP-LS Producers R2 and R3. Here R2 and R3 provide a redundant topology feed via BGP-LS to R0. Normally, R0 would receive two identical copies of all the Link-State NLRIs from both R2 and R3 and it would pick one of them (say R2) based on the standard BGP best path decision process.

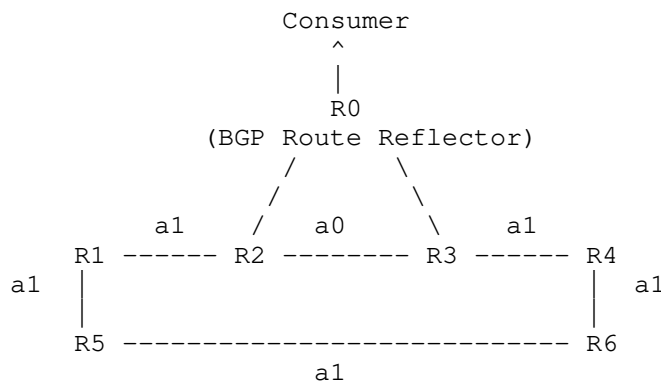


Figure 31: Incorrect Reporting due to BGP Path Selection

Now, R5 will remove the link 5-6 from its Router LSA and this updated LSA is available at R2. R2 also has a stale copy of R6's Router LSA which still has the link 6-5 in it. Based on this view in its LSDB, R2 will advertise only the half-link 6-5 that it derives from R6's stale Router LSA.

At the same time, R6 has removed the link 6-5 from its Router LSA and this updated LSA is available at R3. Similarly, R3 also has a stale copy of R5's Router LSA having the link 5-6 in it. Based on it's LSDB, R3 will advertise only the half-link 5-6 that it has derived from R5's stale Router LSA.

Now, the BGP-LS Consumer receives both the Link NLRIs corresponding to the half-links from R2 and R3 via R0. When viewed together, it would not detect or realize that the area 1 is actually partitioned.

Also if R2 continues to report Link-State NLRIs corresponding to the stale copy of Router LSA of R4 and R6 nodes then R0 would prefer them over the valid Link-State NLRIs for R4 and R6 that it is receiving from R3 based on its BGP decision process. This would result in the BGP-LS Consumer getting stale and inaccurate topology information. This problems scenario is avoided if R2 were to not advertise the link-state information corresponding to R4 and R6 and if R3 were to not advertise similarly for R1 and R5.

A BGP-LS Producer MUST withdraw all link-state objects advertised by it in BGP when the node that originated its corresponding LSP/LSAs is determined to have become unreachable in the IGP and it MUST re-advertise those link-state objects only after that node becomes reachable again in the IGP domain.

4.8. Router-ID Anchoring Example: ISO Pseudonode

Encoding of a broadcast LAN in IS-IS provides a good example of how Router-IDs are encoded. Consider Figure 32. This represents a Broadcast LAN between a pair of routers. The "real" (non-pseudonode) routers have both an IPv4 Router-ID and IS-IS Node-ID. The pseudonode does not have an IPv4 Router-ID. Node1 is the DIS for the LAN. Two unidirectional links (Node1, Pseudonode1) and (Pseudonode1, Node2) are being generated.

The Link NLRI of (Node1, Pseudonode1) is encoded as follows. The IGP Router-ID TLV of the local Node Descriptor is 6 octets long and contains the ISO-ID of Node1, 1920.0000.2001. The IGP Router-ID TLV of the remote Node Descriptor is 7 octets long and contains the ISO-ID of Pseudonode1, 1920.0000.2001.02. The BGP-LS attribute of this link contains one local IPv4 Router-ID TLV (TLV type 1028) containing 192.0.2.1, the IPv4 Router-ID of Node1.

The Link NLRI of (Pseudonode1, Node2) is encoded as follows. The IGP Router-ID TLV of the local Node Descriptor is 7 octets long and contains the ISO-ID of Pseudonode1, 1920.0000.2001.02. The IGP Router-ID TLV of the remote Node Descriptor is 6 octets long and contains the ISO-ID of Node2, 1920.0000.2002. The BGP-LS attribute of this link contains one remote IPv4 Router-ID TLV (TLV type 1030) containing 192.0.2.2, the IPv4 Router-ID of Node2.

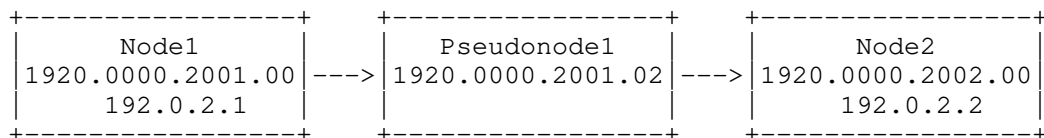


Figure 32: IS-IS Pseudonodes

4.9. Router-ID Anchoring Example: OSPF Pseudonode

Encoding of a broadcast LAN in OSPF provides a good example of how Router-IDs and local Interface IPs are encoded. Consider Figure 33. This represents a Broadcast LAN between a pair of routers. The "real" (non-pseudonode) routers have both an IPv4 Router-ID and an Area Identifier. The pseudonode does have an IPv4 Router-ID, an IPv4 Interface Address (for disambiguation), and an OSPF Area. Node1 is the DR for the LAN; hence, its local IP address 10.1.1.1 is used as both the Router-ID and Interface IP for the pseudonode keys. Two unidirectional links, (Node1, Pseudonode1) and (Pseudonode1, Node2), are being generated.

The Link NLRI of (Node1, Pseudonode1) is encoded as follows:

- o Local Node Descriptor
 - TLV #515: IGP Router-ID: 11.11.11.11
 - TLV #514: OSPF Area-ID: ID:0.0.0.0
- o Remote Node Descriptor
 - TLV #515: IGP Router-ID: 11.11.11.11:10.1.1.1
 - TLV #514: OSPF Area-ID: ID:0.0.0.0

The Link NLRI of (Pseudonode1, Node2) is encoded as follows:

- o Local Node Descriptor
 - TLV #515: IGP Router-ID: 11.11.11.11:10.1.1.1
 - TLV #514: OSPF Area-ID: ID:0.0.0.0
- o Remote Node Descriptor
 - TLV #515: IGP Router-ID: 33.33.33.34
 - TLV #514: OSPF Area-ID: ID:0.0.0.0

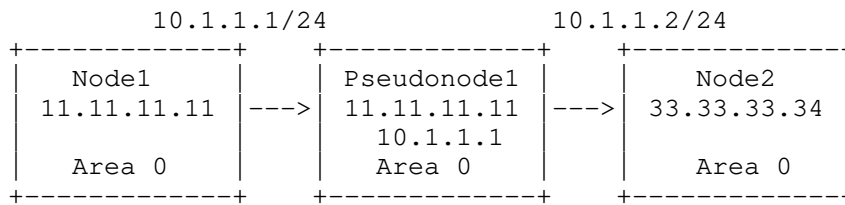


Figure 33: OSPF Pseudonodes

The LAN subnet 10.1.1.0/24 is not included in the Router LSA of Node1 or Node2. The Network LSA for this LAN advertised by the DR Node1 contains the subnet mask for the LAN along with the DR address. A Prefix NLRI corresponding to the LAN subnet is advertised with the Pseudonode1 used as the Local node using the DR address and the subnet mask from the Network LSA.

4.10. Router-ID Anchoring Example: OSPFv2 to IS-IS Migration

Graceful migration from one IGP to another requires coordinated operation of both protocols during the migration period. Such a coordination requires identifying a given physical link in both IGPs. The IPv4 Router-ID provides that "glue", which is present in the Node Descriptors of the OSPF Link NLRI and in the link attribute of the IS-IS Link NLRI.

Consider a point-to-point link between two routers, A and B, that initially were OSPFv2-only routers and then IS-IS is enabled on them. Node A has IPv4 Router-ID and ISO-ID; node B has IPv4 Router-ID, IPv6 Router-ID, and ISO-ID. Each protocol generates one Link NLRI for the link (A, B), both of which are carried by BGP-LS. The OSPFv2 Link NLRI for the link is encoded with the IPv4 Router-ID of nodes A and B in the local and remote Node Descriptors, respectively. The IS-IS Link NLRI for the link is encoded with the ISO-ID of nodes A and B in the local and remote Node Descriptors, respectively. In addition, the BGP-LS attribute of the IS-IS Link NLRI contains the TLV type 1028 containing the IPv4 Router-ID of node A, TLV type 1030 containing the IPv4 Router-ID of node B, and TLV type 1031 containing the IPv6 Router-ID of node B. In this case, by using IPv4 Router-ID, the link (A, B) can be identified in both the IS-IS and OSPF protocol.

5. Link to Path Aggregation

Distribution of all links available in the global Internet is certainly possible; however, it is not desirable from a scaling and privacy point of view. Therefore, an implementation may support a link to path aggregation. Rather than advertising all specific links

of a domain, an ASBR may advertise an "aggregate link" between a non-adjacent pair of nodes. The "aggregate link" represents the aggregated set of link properties between a pair of non-adjacent nodes. The actual methods to compute the path properties (of bandwidth, metric, etc.) are outside the scope of this document. The decision whether to advertise all specific links or aggregated links is an operator's policy choice. To highlight the varying levels of exposure, the following deployment examples are discussed.

5.1. Example: No Link Aggregation

Consider Figure 34. Both AS1 and AS2 operators want to protect their inter-AS {R1, R3}, {R2, R4} links using RSVP-FRR LSPs. If R1 wants to compute its link-protection LSP to R3, it needs to "see" an alternate path to R3. Therefore, the AS2 operator exposes its topology. All BGP-TE-enabled routers in AS1 "see" the full topology of AS2 and therefore can compute a backup path. Note that the computing router decides if the direct link between {R3, R4} or the {R4, R5, R3} path is used.

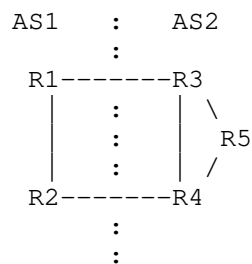


Figure 34: No Link Aggregation

5.2. Example: ASBR to ASBR Path Aggregation

The brief difference between the "no-link aggregation" example and this example is that no specific link gets exposed. Consider Figure 35. The only link that gets advertised by AS2 is an "aggregate" link between R3 and R4. This is enough to tell AS1 that there is a backup path. However, the actual links being used are hidden from the topology.

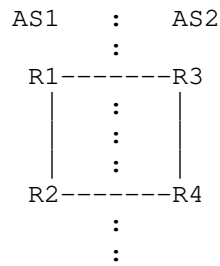


Figure 35: ASBR Link Aggregation

5.3. Example: Multi-AS Path Aggregation

Service providers in control of multiple ASes may even decide to not expose their internal inter-AS links. Consider Figure 36. AS3 is modeled as a single node that connects to the border routers of the aggregated domain.

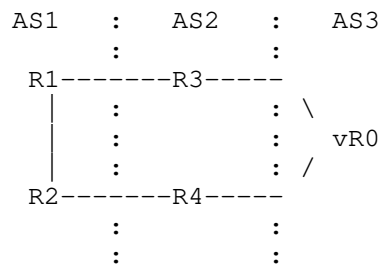


Figure 36: Multi-AS Aggregation

6. IANA Considerations

IANA has assigned address family number 16388 (BGP-LS) in the "Address Family Numbers" registry with [RFC7752] as a reference.

IANA has assigned SAFI values 71 (BGP-LS) and 72 (BGP-LS-VPN) in the "SAFI Values" sub-registry under the "Subsequent Address Family Identifiers (SAFI) Parameters" registry.

IANA has assigned value 29 (BGP-LS Attribute) in the "BGP Path Attributes" sub-registry under the "Border Gateway Protocol (BGP) Parameters" registry.

IANA has created a new "Border Gateway Protocol - Link State (BGP-LS) Parameters" registry at <http://www.iana.org/assignments/bgp-ls-parameters>. All of the following registries are BGP-LS specific and are accessible under this registry:

- o "BGP-LS NLRI-Types" registry

Value 0 is reserved. The maximum value is 65535. The range 65000-65535 is for Private Use. The registry has been populated with the values shown in Table 1. Allocations within the registry require documentation of the proposed use of the allocated value (Specification Required) and approval by the Designated Expert assigned by the IESG (see [RFC8126]).

- o "BGP-LS Protocol-IDs" registry

Value 0 is reserved. The maximum value is 255. The range 200-255 is for Private Use. The registry has been populated with the values shown in Table 2. Allocations within the registry require documentation of the proposed use of the allocated value (Specification Required) and approval by the Designated Expert assigned by the IESG (see [RFC8126]).

- o "BGP-LS Well-Known Instance-IDs" registry

This registry was setup via [RFC7752] and is no longer required. It may be retained as deprecated.

- o "BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs" registry

Values 0-255 are reserved. Values 256-65535 will be used for code points. The range 65000-65535 is for Private Use. The registry has been populated with the values shown in Table 12. Allocations within the registry require documentation of the proposed use of the allocated value (Specification Required) and approval by the Designated Expert assigned by the IESG (see [RFC8126]).

6.1. Guidance for Designated Experts

In all cases of review by the Designated Expert (DE) described here, the DE is expected to ascertain the existence of suitable documentation (a specification) as described in [RFC8126] and to verify that the document is permanently and publicly available. The DE is also expected to check the clarity of purpose and use of the requested code points. Last, the DE must verify that any specification produced in the IETF that requests one of these code points has been made available for review by the IDR working group and that any specification produced outside the IETF does not conflict with work that is active or already published within the IETF.

7. Manageability Considerations

This section is structured as recommended in [RFC5706].

7.1. Operational Considerations

7.1.1. Operations

Existing BGP operational procedures apply. No new operation procedures are defined in this document. It is noted that the NLRI information present in this document carries purely application-level data that has no immediate impact on the corresponding forwarding state computed by BGP. As such, any churn in reachability information has a different impact than regular BGP updates, which need to change the forwarding state for an entire router. It is expected that the distribution of this NLRI SHOULD be handled by dedicated route reflectors in most deployments providing a level of isolation and fault containment between different NLRI types. In the event of dedicated route reflectors not being available, other alternate mechanisms like separation of BGP instances or separate BGP sessions (e.g. using different addresses for peering) for Link-State information distribution SHOULD be used.

7.1.2. Installation and Initial Setup

Configuration parameters defined in Section 7.2.3 SHOULD be initialized to the following default values:

- o The Link-State NLRI capability is turned off for all neighbors.
- o The maximum rate at which Link-State NLRIs will be advertised/withdrawn from neighbors is set to 200 updates per second.

7.1.3. Migration Path

The proposed extension is only activated between BGP peers after capability negotiation. Moreover, the extensions can be turned on/off on an individual peer basis (see Section 7.2.3), so the extension can be gradually rolled out in the network.

7.1.4. Requirements on Other Protocols and Functional Components

The protocol extension defined in this document does not put new requirements on other protocols or functional components.

7.1.5. Impact on Network Operation

Frequency of Link-State NLRI updates could interfere with regular BGP prefix distribution. A network operator MAY use a dedicated Route-Reflector infrastructure to distribute Link-State NLRIs.

Distribution of Link-State NLRIs SHOULD be limited to a single admin domain, which can consist of multiple areas within an AS or multiple ASes.

7.1.6. Verifying Correct Operation

Existing BGP procedures apply. In addition, an implementation SHOULD allow an operator to:

- o List neighbors with whom the speaker is exchanging Link-State NLRIs.

7.2. Management Considerations

7.2.1. Management Information

The IDR working group has documented and continues to document parts of the Management Information Base and YANG models for managing and monitoring BGP speakers and the sessions between them. It is currently believed that the BGP session running BGP-LS is not substantially different from any other BGP session and can be managed using the same data models.

7.2.2. Fault Management

This section describes the fault management actions, as described in [RFC7606], that are to be performed for handling of BGP update messages for BGP-LS.

A Link-State NLRI MUST NOT be considered as malformed or invalid based on the inclusion/exclusion of TLVs or contents of the TLV fields (i.e. semantic errors), as described in Section 4.1 and Section 4.2.

A BGP-LS Speaker MUST perform the following syntactic validation of the Link-State NLRI to determine if it is malformed.

- o Does the sum of all TLVs found in the BGP MP_REACH_NLRI attribute correspond to the BGP MP_REACH_NLRI length?
- o Does the sum of all TLVs found in the BGP MP_UNREACH_NLRI attribute correspond to the BGP MP_UNREACH_NLRI length?

- o Does the sum of all TLVs found in a Link-State NLRI correspond to the Total NLRI Length field of all its Descriptors?
- o Is the length of the TLVs and, when the TLV is recognized then, its sub-TLVs in the NLRI valid?
- o Has the syntactic correctness of the NLRI fields been verified as per [RFC7606]?
- o Has the rule regarding ordering of TLVs been followed as described in Section 4.1?

When the error determined allows for the router to skip the malformed NLRI(s) and continue processing of the rest of the update message (e.g. when the TLV ordering rule is violated), 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 BGP-LS are being advertised over the same session. Alternately, the router MUST perform 'session reset' when the session is only being used for BGP-LS or when it 'AFI/SAFI disable' action is not possible.

A BGP-LS Attribute MUST NOT be considered as malformed or invalid based on the inclusion/exclusion of TLVs or contents of the TLV fields (i.e. semantic errors), as described in Section 4.1 and Section 4.3.

A BGP-LS Speaker MUST perform the following syntactic validation of the BGP-LS Attribute to determine if it is malformed.

- o Does the sum of all TLVs found in the BGP-LS Attribute correspond to the BGP-LS Attribute length?
- o Has the syntactic correctness of the Attributes (including BGP-LS Attribute) been verified as per [RFC7606]?
- o Is the length of each TLV and, when the TLV is recognized then, its sub-TLVs in the BGP-LS Attribute valid?

When the error determined allows for the router to skip the malformed BGP-LS Attribute and continue processing of the rest of the update message (e.g. when the BGP-LS Attribute length and the total Path Attribute Length are correct but some TLV/sub-TLV length within the BGP-LS Attribute is invalid), then it MUST handle such malformed BGP-LS Attribute as 'Attribute Discard'. In other cases, where the error in the BGP-LS Attribute encoding results in the inability to process

the BGP update message then the handling is the same as described above for the malformed NLRI.

Note that the 'Attribute Discard' action results in the loss of all TLVs in the BGP-LS Attribute and not the removal of a specific malformed TLV. The removal of specific malformed TLVs may give a wrong indication to a BGP-LS Consumer of that specific information being deleted or not available.

When a BGP Speaker receives an update message with Link-State NLRI(s) in the MP_REACH_NLRI but without the BGP-LS Attribute, it is most likely an indication that a BGP Speaker preceding it has performed the 'Attribute Discard' fault handling. An implementation SHOULD preserve and propagate the Link-State NLRIs in such an update message so that the BGP-LS Consumers can detect the loss of link-state information for that object and not assume its deletion/withdraw. This also makes it possible for a network operator to trace back to the BGP-LS Propagator which actually detected a fault with the BGP-LS Attribute.

An implementation SHOULD log an error for any errors found during syntax validation for further analysis.

A BGP-LS Propagator SHOULD NOT perform semantic validation of the Link-State NLRI or the BGP-LS Attribute to determine if it is malformed or invalid. Some types of semantic validation that are not to be performed by a BGP-LS Propagator are as follows (and this is not to be considered as an exhaustive list):

- o is a mandatory TLV present or not?
- o is the length of a fixed length TLV correct or the length of a variable length TLV a valid/missible?
- o are the values of TLV fields valid or permissible?
- o are the inclusion and use of TLVs/sub-TLVs with specific Link-State NLRI types valid?

Each TLV MAY indicate the valid and permissible values and their semantics that can to be used only by a BGP-LS Consumer for its semantic validation. However, the handling of any errors may be specific to the particular application and outside the scope of this document. A BGP-LS Consumer should ignore unrecognized and unexpected TLV types in both the NLRI and BGP-LS Attribute portions and not consider their presence as an error.

7.2.3. Configuration Management

An implementation SHOULD allow the operator to specify neighbors to which Link-State NLRIs will be advertised and from which Link-State NLRIs will be accepted.

An implementation SHOULD allow the operator to specify the maximum rate at which Link-State NLRIs will be advertised/withdrawn from neighbors.

An implementation SHOULD allow the operator to specify the maximum number of Link-State NLRIs stored in a router's Routing Information Base (RIB).

An implementation SHOULD allow the operator to create abstracted topologies that are advertised to neighbors and create different abstractions for different neighbors.

An implementation SHOULD allow the operator to configure a 64-bit Instance-ID.

An implementation SHOULD allow the operator to configure ASN and BGP-LS identifiers (refer Section 4.2.1.4).

An implementation SHOULD allow the operator to configure the maximum size of the BGP-LS Attribute that may be used on a BGP-LS Producer.

7.2.4. Accounting Management

Not Applicable.

7.2.5. Performance Management

An implementation SHOULD provide the following statistics:

- o Total number of Link-State NLRI updates sent/received
- o Number of Link-State NLRI updates sent/received, per neighbor
- o Number of errored received Link-State NLRI updates, per neighbor
- o Total number of locally originated Link-State NLRIs

These statistics should be recorded as absolute counts since system or session start time. An implementation MAY also enhance this information by recording peak per-second counts in each case.

7.2.6. Security Management

An operator SHOULD define an import policy to limit inbound updates as follows:

- o Drop all updates from peers that are only serving BGP-LS Consumers.

An implementation MUST have the means to limit inbound updates.

8. TLV/Sub-TLV Code Points Summary

This section contains the global table of all TLVs/sub-TLVs defined in this document.

| TLV Code Point | Description | IS-IS TLV/ Sub-TLV | Reference (RFC/Section) |
|----------------|--------------------------------|-----------------------|----------------------------|
| 256 | Local Node Descriptors | --- | Section 4.2.1.2 |
| 257 | Remote Node Descriptors | --- | Section 4.2.1.3 |
| 258 | Link Local/Remote Identifiers | 22/4 | [RFC5307]/1.1 |
| 259 | IPv4 interface address | 22/6 | [RFC5305]/3.2 |
| 260 | IPv4 neighbor address | 22/8 | [RFC5305]/3.3 |
| 261 | IPv6 interface address | 22/12 | [RFC6119]/4.2 |
| 262 | IPv6 neighbor address | 22/13 | [RFC6119]/4.3 |
| 263 | Multi-Topology ID | --- | Section 4.2.2.1 |
| 264 | OSPF Route Type | --- | Section 4.2.3 |
| 265 | IP Reachability Information | --- | Section 4.2.3 |
| 512 | Autonomous System | --- | Section 4.2.1.4 |
| 513 | BGP-LS Identifier (deprecated) | --- | Section 4.2.1.4 |
| 514 | OSPF Area-ID | --- | Section 4.2.1.4 |
| 515 | IGP Router-ID | --- | Section 4.2.1.4 |
| 1024 | Node Flag Bits | --- | Section 4.3.1.1 |
| 1025 | Opaque Node Attribute | --- | Section 4.3.1.5 |
| 1026 | Node Name | variable | Section 4.3.1.3 |
| 1027 | IS-IS Area Identifier | variable | Section 4.3.1.2 |

| | | | |
|------|--------------------------------|---------|-----------------|
| 1028 | IPv4 Router-ID of Local Node | 134/--- | [RFC5305]/4.3 |
| 1029 | IPv6 Router-ID of Local Node | 140/--- | [RFC6119]/4.1 |
| 1030 | IPv4 Router-ID of Remote Node | 134/--- | [RFC5305]/4.3 |
| 1031 | IPv6 Router-ID of Remote Node | 140/--- | [RFC6119]/4.1 |
| 1088 | Administrative group (color) | 22/3 | [RFC5305]/3.1 |
| 1089 | Maximum link bandwidth | 22/9 | [RFC5305]/3.4 |
| 1090 | Max. reservable link bandwidth | 22/10 | [RFC5305]/3.5 |
| 1091 | Unreserved bandwidth | 22/11 | [RFC5305]/3.6 |
| 1092 | TE Default Metric | 22/18 | Section 4.3.2.3 |
| 1093 | Link Protection Type | 22/20 | [RFC5307]/1.2 |
| 1094 | MPLS Protocol Mask | --- | Section 4.3.2.2 |
| 1095 | IGP Metric | --- | Section 4.3.2.4 |
| 1096 | Shared Risk Link Group | --- | Section 4.3.2.5 |
| 1097 | Opaque Link Attribute | --- | Section 4.3.2.6 |
| 1098 | Link Name | --- | Section 4.3.2.7 |
| 1152 | IGP Flags | --- | Section 4.3.3.1 |
| 1153 | IGP Route Tag | --- | [RFC5130] |
| 1154 | IGP Extended Route Tag | --- | [RFC5130] |
| 1155 | Prefix Metric | --- | [RFC5305] |
| 1156 | OSPF Forwarding Address | --- | [RFC2328] |
| 1157 | Opaque Prefix Attribute | --- | Section 4.3.3.6 |

Table 12: Summary Table of TLV/Sub-TLV Code Points

9. Security Considerations

Procedures and protocol extensions defined in this document do not affect the BGP security model. 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.

In the context of the BGP peerings associated with this document, a BGP speaker MUST NOT accept updates from a peer that is only

providing information to a BGP-LS Consumer. That is, a participating BGP speaker should be aware of the nature of its relationships for link-state relationships and should protect itself from peers sending updates that either represent erroneous information feedback loops or are false input. Such protection can be achieved by manual configuration of consumer peers at the BGP speaker.

An operator SHOULD employ a mechanism to protect a BGP speaker against DDoS attacks from BGP-LS Consumers. The principal attack a consumer may apply is to attempt to start multiple sessions either sequentially or simultaneously. Protection can be applied by imposing rate limits.

Additionally, it may be considered that the export of link-state and TE information as described in this document constitutes a risk to confidentiality of mission-critical or commercially sensitive information about the network. BGP peerings are not automatic and require configuration; thus, it is the responsibility of the network operator to ensure that only trusted consumers are configured to receive such information.

10. Contributors

We would like to thank Robert Varga for the significant contribution he gave to RFC7752.

11. Acknowledgements

This document update to the BGP-LS specification [RFC7752] is a result of feedback and inputs from the discussions in the IDR working group. It also incorporates certain details and clarifications based on implementation and deployment experience with BGP-LS.

Cengiz Alaettinoglu and Parag Amritkar brought forward the need to clarify the advertisement of LAN subnet for OSPF.

We would like to thank Balaji Rajagopalan, Srihari Sangli and Shraddha Hegde for their review and feedback on this document.

We would like to thank Nischal Sheth, Alia Atlas, David Ward, Derek Yeung, Murtuza Lightwala, John Scudder, Kaliraj Vairavakkalai, Les Ginsberg, Liem Nguyen, Manish Bhardwaj, Matt Miller, Mike Shand, Peter Psenak, Rex Fernando, Richard Woundy, Steven Luong, Tamas Mondal, Waqas Alam, Vipin Kumar, Naiming Shen, Carlos Pignataro, Balaji Rajagopalan, Yakov Rekhter, Alvaro Retana, Barry Leiba, and Ben Campbell for their comments on RFC7752.

12. References

12.1. Normative References

- [I-D.ietf-idr-bgp-extended-messages]
Bush, R., Patel, K., and D. Ward, "Extended Message support for BGP", draft-ietf-idr-bgp-extended-messages-33 (work in progress), July 2019.
- [ISO10589]
International Organization for Standardization, "Intermediate System to Intermediate System intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473)", ISO/IEC 10589, November 2002.
- [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>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC2545] Marques, P. and F. Dupont, "Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing", RFC 2545, DOI 10.17487/RFC2545, March 1999, <<https://www.rfc-editor.org/info/rfc2545>>.
- [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>>.
- [RFC4203] Kompella, K., Ed. and Y. Rekhter, Ed., "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4203, DOI 10.17487/RFC4203, October 2005, <<https://www.rfc-editor.org/info/rfc4203>>.

- [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>>.
- [RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, DOI 10.17487/RFC4915, June 2007, <<https://www.rfc-editor.org/info/rfc4915>>.
- [RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <<https://www.rfc-editor.org/info/rfc5036>>.
- [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>>.
- [RFC5130] Previdi, S., Shand, M., Ed., and C. Martin, "A Policy Control Mechanism in IS-IS Using Administrative Tags", RFC 5130, DOI 10.17487/RFC5130, February 2008, <<https://www.rfc-editor.org/info/rfc5130>>.
- [RFC5301] McPherson, D. and N. Shen, "Dynamic Hostname Exchange Mechanism for IS-IS", RFC 5301, DOI 10.17487/RFC5301, October 2008, <<https://www.rfc-editor.org/info/rfc5301>>.
- [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>>.
- [RFC5307] Kompella, K., Ed. and Y. Rekhter, Ed., "IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 5307, DOI 10.17487/RFC5307, October 2008, <<https://www.rfc-editor.org/info/rfc5307>>.
- [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>>.

- [RFC5642] Venkata, S., Harwani, S., Pignataro, C., and D. McPherson, "Dynamic Hostname Exchange Mechanism for OSPF", RFC 5642, DOI 10.17487/RFC5642, August 2009, <<https://www.rfc-editor.org/info/rfc5642>>.
- [RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <<https://www.rfc-editor.org/info/rfc5890>>.
- [RFC6119] Harrison, J., Berger, J., and M. Bartlett, "IPv6 Traffic Engineering in IS-IS", RFC 6119, DOI 10.17487/RFC6119, February 2011, <<https://www.rfc-editor.org/info/rfc6119>>.
- [RFC6549] Lindem, A., Roy, A., and S. Mirtorabi, "OSPFv2 Multi-Instance Extensions", RFC 6549, DOI 10.17487/RFC6549, March 2012, <<https://www.rfc-editor.org/info/rfc6549>>.
- [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>>.
- [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>>.
- [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>>.
- [RFC8202] Ginsberg, L., Previdi, S., and W. Henderickx, "IS-IS Multi-Instance", RFC 8202, DOI 10.17487/RFC8202, June 2017, <<https://www.rfc-editor.org/info/rfc8202>>.

12.2. Informative References

- [RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <<https://www.rfc-editor.org/info/rfc1918>>.

- [RFC4272] Murphy, S., "BGP Security Vulnerabilities Analysis", RFC 4272, DOI 10.17487/RFC4272, January 2006, <<https://www.rfc-editor.org/info/rfc4272>>.
- [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>>.
- [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <<https://www.rfc-editor.org/info/rfc4655>>.
- [RFC5073] Vasseur, J., Ed. and J. Le Roux, Ed., "IGP Routing Protocol Extensions for Discovery of Traffic Engineering Node Capabilities", RFC 5073, DOI 10.17487/RFC5073, December 2007, <<https://www.rfc-editor.org/info/rfc5073>>.
- [RFC5152] Vasseur, JP., Ed., Ayyangar, A., Ed., and R. Zhang, "A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs)", RFC 5152, DOI 10.17487/RFC5152, February 2008, <<https://www.rfc-editor.org/info/rfc5152>>.
- [RFC5316] Chen, M., Zhang, R., and X. Duan, "ISIS Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5316, DOI 10.17487/RFC5316, December 2008, <<https://www.rfc-editor.org/info/rfc5316>>.
- [RFC5392] Chen, M., Zhang, R., and X. Duan, "OSPF Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5392, DOI 10.17487/RFC5392, January 2009, <<https://www.rfc-editor.org/info/rfc5392>>.
- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, DOI 10.17487/RFC5693, October 2009, <<https://www.rfc-editor.org/info/rfc5693>>.
- [RFC5706] Harrington, D., "Guidelines for Considering Operations and Management of New Protocols and Protocol Extensions", RFC 5706, DOI 10.17487/RFC5706, November 2009, <<https://www.rfc-editor.org/info/rfc5706>>.

- [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>>.
- [RFC7285] Alimi, R., Ed., Penno, R., Ed., Yang, Y., Ed., Kiesel, S., Previdi, S., Roome, W., Shalunov, S., and R. Woundy, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, DOI 10.17487/RFC7285, September 2014, <<https://www.rfc-editor.org/info/rfc7285>>.
- [RFC7770] Lindem, A., Ed., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 7770, DOI 10.17487/RFC7770, February 2016, <<https://www.rfc-editor.org/info/rfc7770>>.

Appendix A. Changes from RFC 7752

This section lists the high-level changes from RFC 7752 and provides reference to the document sections wherein those have been introduced.

1. Update the Figure 1 in Section 1 and added Section 3 to illustrate the different roles of a BGP implementation in conveying link-state information.
2. In Section 4.1, clarification about the TLV handling aspects that are applicable to both the NLRI and BGP-LS Attribute parts and those that are applicable only for the NLRI portion. An implementation may have missed the part about handling of unrecognized TLV and so, based on [RFC7606] guidelines, might discard the unknown NLRI types. This aspect is now unambiguously clarified in Section 4.2. Also, the ascending order of TLVs in the BGP-LS Attribute is not necessary.
3. Clarification of mandatory and optional TLVs in both NLRI and BGP-LS Attribute portions all through the document.
4. Handling of the growth of the BGP-LS Attribute is covered in Section 4.3.
5. Clarification on the use of Identifier field in the Link-State NLRI in Section 4.2 is provided. It was defined ambiguously to refer to only multi-instance IGP on a single link while it can also be used for multiple IGP protocol instances on a router. The IANA registry is accordingly being removed.

6. The BGP-LS Identifier TLV in the Node Descriptors has been deprecated. Its use was not well specified by [RFC7752] and there has been some amount of confusion between implementators on its usage for identification of IGP domains as against the use of the Identifier doing the same functionality as the Instance-ID when running multiple instances of IGP routing protocols.
7. Moved MT-ID TLV from the Node Descriptor section to under the Link Descriptor section since it is not a Node Descriptor sub-TLV. Also fixed the ambiguity in the encoding of OSPF MT-ID in this TLV. MT-ID TLV use is now elevated to SHOULD when it is enabled in the underlying IGP.
8. Update the usage of OSPF Route Type TLV to mandate its use for OSPF prefixes in Section 4.2.3.1 since this is required for segregation of intra-area prefixes that are used to reach a node (e.g. a loopback) from other types of inter-area and external prefixes.
9. Updated the Node Name TLV in Section 4.3.1.3 with the OSPF specification.
10. Clarified the advertisement of the prefix corresponding to the LAN segment in an OSPF network in Section 4.9.
11. Introduced Private Use TLV code point space and specified their encoding in Section 4.4.
12. Introduced Section 4.7 where issues related to consistency of reporting IGP link-state along with their solutions are covered.
13. Handling of large size of BGP-LS Attribute with growth in BGP-LS information is explained in Section 4.3 along with mitigation of errors arising out of it.
14. Added recommendation for isolation of BGP-LS sessions from other BGP route exchange to avoid errors and faults in BGP-LS affecting the normal BGP routing.
15. Updated the Fault Management section with detailed rules based on the role in the BGP-LS information propagation flow.

Authors' Addresses

Ketan Talaulikar (editor)
Cisco Systems
India

Email: ketant@cisco.com

Hannes Gredler
Rtbrick

Email: hannes@rtbrick.com

Jan Medved
Cisco Systems, Inc.
170, West Tasman Drive
San Jose, CA 95134
US

Email: jmedved@cisco.com

Stefano Previdi
Individual Contributor
Rome
Italy

Email: stefano@previdi.net

Adrian Farrel
Old Dog Consulting

Email: adrian@olddog.co.uk

Saikat Ray
Individual Contributor

Email: raysaikat@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2020

Z. Li
Huawei
L. Ou
Y. Luo
China Telcom Co., Ltd.
S. Lu
Tencent
H. Chen
Futurewei
S. Zhuang
H. Wang
Huawei
July 7, 2019

BGP Extensions for Routing Policy Distribution (RPD)
draft-li-idr-flowspec-rpd-05

Abstract

It is hard to adjust traffic and optimize traffic paths on a traditional IP network from time to time through manual configurations. It is desirable to have an automatic mechanism for setting up routing policies, which adjust traffic and optimize traffic paths automatically. This document describes BGP Extensions for Routing Policy Distribution (BGP RPD) to support this.

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 January 8, 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. Terminology | 3 |
| 3. Problem Statements | 3 |
| 3.1. Inbound Traffic Control | 3 |
| 3.2. Outbound Traffic Control | 4 |
| 4. Protocol Extensions | 5 |
| 4.1. Using a New AFI and SAFI | 5 |
| 4.2. BGP Wide Community | 6 |
| 4.2.1. New Wide Community Atoms | 6 |
| 4.3. Capability Negotiation | 12 |
| 5. Consideration | 12 |
| 5.1. Route-Policy | 12 |
| 6. Contributors | 13 |
| 7. Security Considerations | 13 |
| 8. Acknowledgements | 14 |
| 9. IANA Considerations | 14 |
| 10. References | 15 |
| 10.1. Normative References | 15 |
| 10.2. Informative References | 16 |
| Authors' Addresses | 16 |

1. Introduction

It is difficult to optimize traffic paths on a traditional IP network because of:

- o Heavy configuration and error prone. Traffic can only be adjusted device by device. All routers that the traffic traverses need to be configured. The configuration workload is heavy. The

operation is not only time consuming but also prone to misconfiguration for Service Providers.

- o Complex. The routing policies used to control network routes are complex, posing difficulties to subsequent maintenance, high maintenance skills are required.

It is desirable to have an automatic mechanism for setting up routing policies, which can simplify the routing policies configuration. This document describes extensions to BGP for Routing Policy Distribution to resolve these issues.

2. Terminology

The following terminology is used in this document.

- o ACL: Access Control List
- o BGP: Border Gateway Protocol
- o FS: Flow Specification
- o PBR: Policy-Based Routing
- o RPD: Routing Policy Distribution
- o VPN: Virtual Private Network

3. Problem Statements

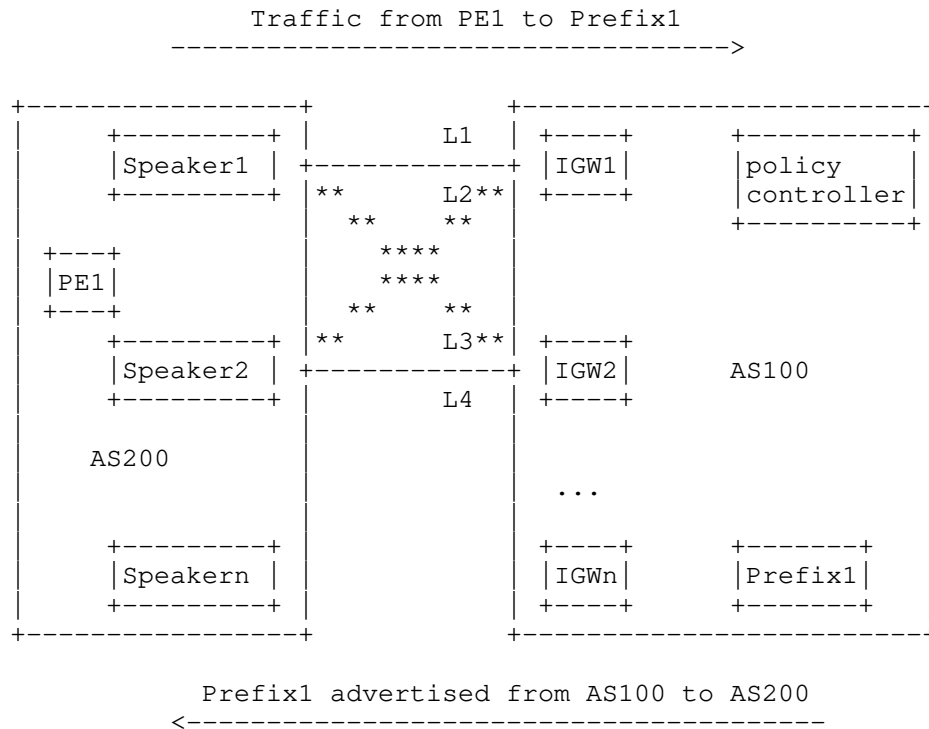
It is obvious that providers have the requirements to adjust their business traffic from time to time because:

- o Business development or network failure introduces link congestion and overload.
- o Network transmission quality is decreased as the result of delay, loss and they need to adjust traffic to other paths.
- o To control OPEX and CPEX, prefer the transit provider with lower price.

3.1. Inbound Traffic Control

In the scenario below, for the reasons above, the provider of AS100 saying P may wish the inbound traffic from AS200 enters AS100 through link L3 instead of the others. Since P doesn't have any

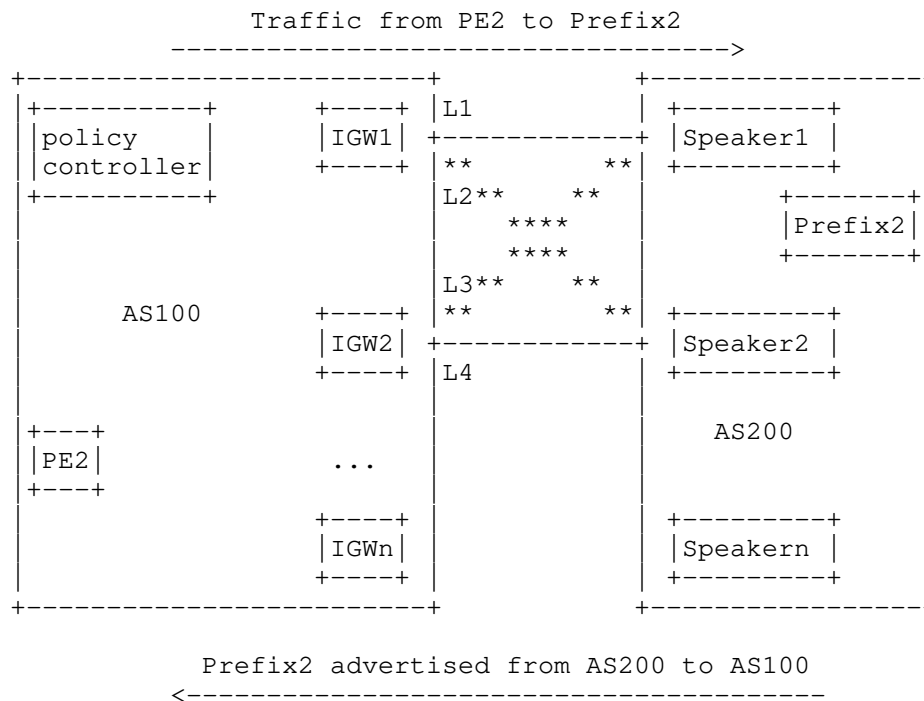
administration over AS200, so there is no way for P to modify the route selection criteria directly.



Inbound Traffic Control case

3.2. Outbound Traffic Control

In the scenario below, the provider of AS100 saying P prefers link L3 for the traffic to the destination Prefix2 among multiple exits and links. This preference can be dynamic and changed frequently because of the reasons above. So the provider P expects an efficient and convenient solution.



Outbound Traffic Control case

4. Protocol Extensions

A solution is proposed to use a new AFI and SAFI with the BGP Wide Community for encoding a routing policy.

4.1. Using a New AFI and SAFI

A new AFI and SAFI are defined: the Routing Policy AFI whose codepoint TBD1 is to be assigned by IANA, and SAFI whose codepoint TBD2 is to be assigned by IANA.

The AFI and SAFI pair uses a new NLRI, which is defined 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
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|  NLRI Length  |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|  Policy Type  |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     Distinguisher (4 octets)                                     |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     Peer IP (4/16 octets)                                     ~
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Where:

NLRI Length: 1 octet represents the length of NLRI.

Policy Type: 1 octet indicates the type of a policy. 1 is for export policy. 2 is for import policy.

Distinguisher: 4 octet value uniquely identifies the policy in the peer.

Peer IP: 4/16 octet value indicates an IPv4/IPv6 peer.

The NLRI containing the Routing Policy is carried in a BGP UPDATE message, which MUST contain the BGP mandatory attributes and MAY also contain some BGP optional attributes.

When receiving a BGP UPDATE message, a BGP speaker processes it only if the peer IP address in the NLRI is the IP address of the BGP speaker or 0.

The content of the Routing Policy is encoded in a BGP Wide Community.

4.2. BGP Wide Community

The BGP wide community is defined in [I-D.ietf-idr-wide-bgp-communities]. It can be used to facilitate the delivery of new network services, and be extended easily for distributing different kinds of routing policies.

4.2.1. New Wide Community Atoms

A wide community Atom is a TLV (or sub-TLV), which may be included in a BGP wide community container (or BGP wide community for short) containing some BGP Wide Community TLVs. Three BGP Wide Community TLVs are defined in [I-D.ietf-idr-wide-bgp-communities], which are BGP Wide Community Target(s) TLV, Exclude Target(s) TLV, and

Parameter(s) TLV. Each of these TLVs comprises a series of Atoms, each of which is a TLV (or sub-TLV). A new wide community Atom is defined for BGP Wide Community Target(s) TLV and a few new Atoms are defined for BGP Wide Community Parameter(s) TLV. For your reference, the format of the TLV is illustrated below:

```

      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                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Value (variable)                                     ~
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Format of Wide Community Atom TLV

A RouteAttr Atom TLV (or RouteAttr TLV/sub-TLV for short) is defined and may be included in a Target TLV. It has the following format.

```

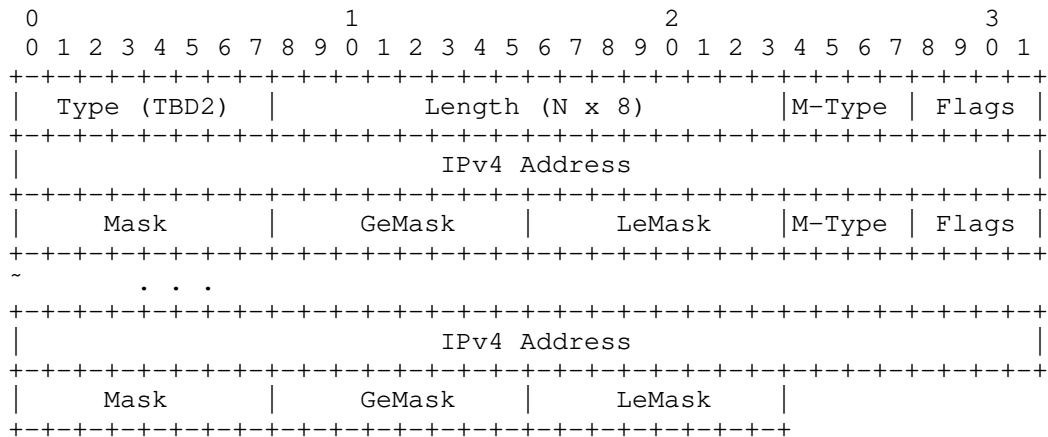
      0                               1                               2                               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|  Type (TBD1)  |                               Length (variable)                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     sub-TLVs                                     ~
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Format of RouteAttr Atom TLV

The Type for RouteAttr is TBD1 (suggested value 48) to be assigned by IANA. In RouteAttr TLV, three sub-TLVs are defined: IP Prefix, AS-Path and Community sub-TLV.

An IP prefix sub-TLV gives matching criteria on IPv4 prefixes. Its format is illustrated below:



Format of IPv4 Prefix sub-TLV

Type: TBD2 (suggested value 1) for IPv4 Prefix is to be assigned by IANA.

Length: N x 8, where N is the number of tuples <M-Type, Flags, IPv4 Address, Mask, GeMask, LeMask>.

M-Type: 4 bits for match types, four of which are defined:

M-Type = 0: Exact match.

M-Type = 1: Match prefix greater and equal to the given masks.

M-Type = 2: Match prefix less and equal to the given masks.

M-Type = 3: Match prefix within the range of the given masks.

Flags: 4 bits. No flags are currently defined.

IPv4 Address: 4 octets for an IPv4 address.

Mask: 1 octet for the mask length.

GeMask: 1 octet for match range, must be less than Mask or be 0.

LeMask: 1 octet for match range, must be greater than Mask or be 0.

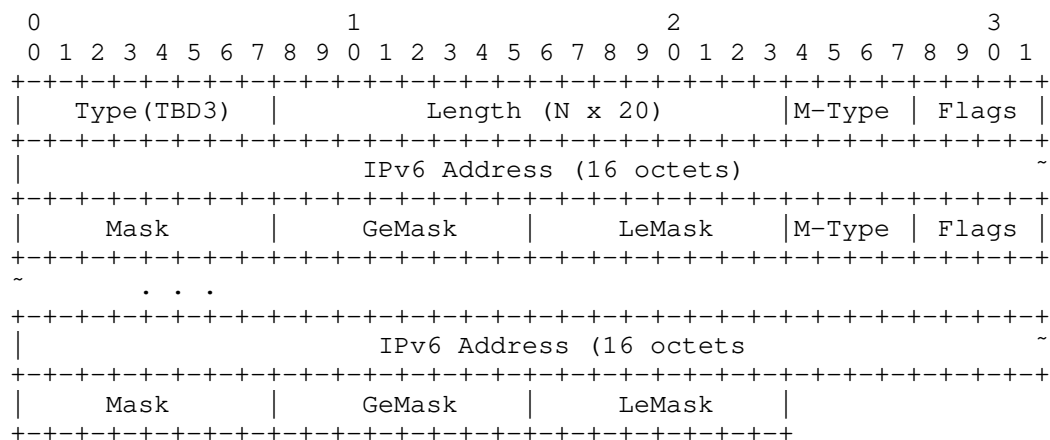
For example, tuple <M-Type=0, Flags=0, IPv4 Address = 1.1.0.0, Mask = 22, GeMask = 0, LeMask = 0> represents an exact IP prefix match for 1.1.0.0/22.

<M-Type=1, Flags=0, IPv4 Address = 16.1.0.0, Mask = 24, GeMask = 24, LeMask = 0> represents match IP prefix 1.1.0.0/24 greater-equal 24.

<M-Type=2, Flags=0, IPv4 Address = 17.1.0.0, Mask = 24, GeMask = 0, LeMask = 26> represents match IP prefix 17.1.0.0/24 less-equal 26.

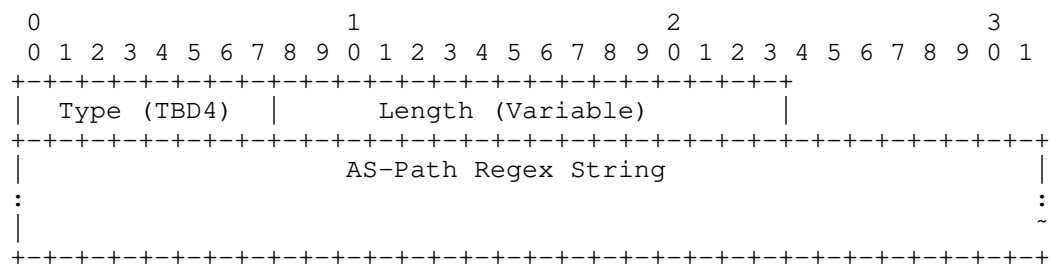
<M-Type=3, Flags=0, IPv4 Address = 18.1.0.0, Mask = 24, GeMask = 24, LeMask = 32> represents match IP prefix 18.1.0.0/24 greater-equal to 24 and less-equal 32.

Similarly, an IPv6 Prefix sub-TLV represents match criteria on IPv6 prefixes. Its format is illustrated below:



Format of IPv6 Prefix sub-TLV

An AS-Path sub-TLV represents a match criteria in a regular expression string. Its format is illustrated below:



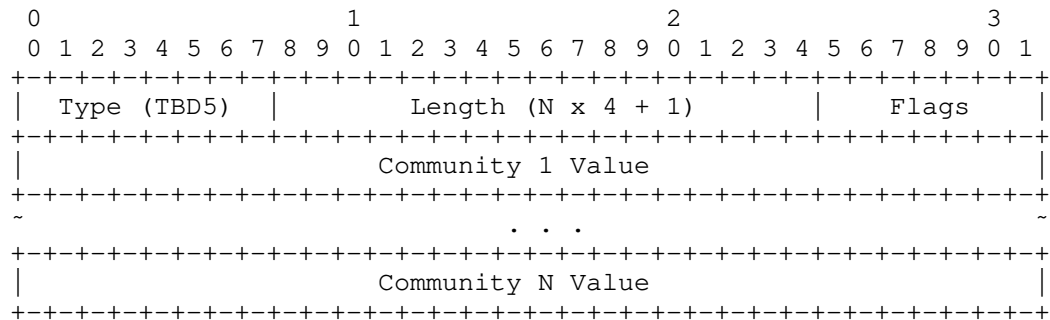
Format of AS Path sub-TLV

Type: TBD4 (suggested value 2) for AS-Path is to be assigned by IANA.

Length: Variable, maximum is 1024.

AS-Path Regex String: AS-Path regular expression string.

A community sub-TLV represents a list of communities to be matched all. Its format is illustrated below:



Format of Community sub-TLV

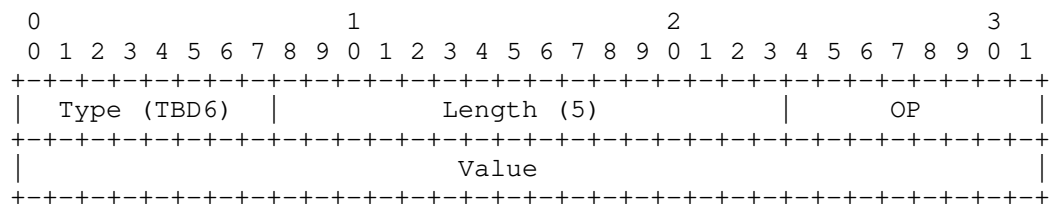
Type: TBD5 (suggested value 3) for Community is to be assigned by IANA.

Length: $N \times 4 + 1$, where N is the number of communities.

Flags: 1 octet. No flags are currently defined.

In Parameter(s) TLV, two action sub-TLVs are defined: MED change sub-TLV and AS-Path change sub-TLV. When the community in the container is MATCH AND SET ATTR, the Parameter(s) TLV includes some of these sub-TLVs. When the community is MATCH AND NOT ADVERTISE, the Parameter(s) TLV's value is empty.

A MED change sub-TLV indicates an action to change the MED. Its format is illustrated below:



Format of MED Change sub-TLV

Type: TBD6 (suggested value 1) for MED Change is to be assigned by IANA.

Length: 5.

OP: 1 octet. Three are defined:

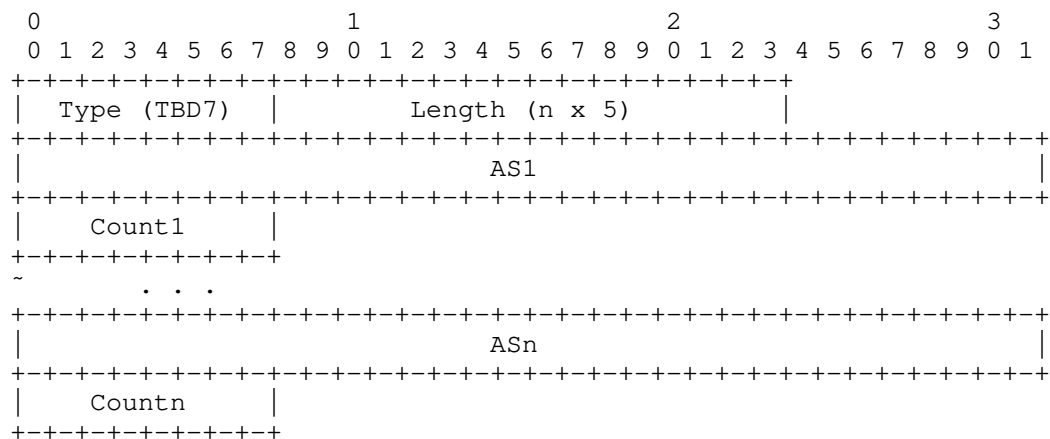
OP = 0: assign the Value to the existing MED.

OP = 1: add the Value to the existing MED. If the sum is greater than the maximum value for MED, assign the maximum value to MED.

OP = 2: subtract the Value from the existing MED. If the existing MED minus the Value is less than 0, assign 0 to MED.

Value: 4 octets.

An AS-Path change sub-TLV indicates an action to change the AS-Path. Its format is illustrated below:



Format of AS-Path Change sub-TLV

Type: TBD7 (suggested value 2) for AS-Path Change is to be assigned by IANA.

Length: n x 5.

ASi: 4 octet. An AS number.

Counti: 1 octet. ASi repeats Counti times.

The sequence of AS numbers are added to the existing AS Path.

4.3. Capability Negotiation

It is necessary to negotiate the capability to support BGP Extensions for Routing Policy Distribution (RPD). The BGP RPD 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 RPD 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 Routing Policies from its peer (value 1), (b) would like to send Routing Policies to its peer (value 2), or (c) both (value 3) for the <AFI, SAFI>.

5. Consideration

5.1. Route-Policy

Routing policies are used to filter routes and control how routes are received and advertised. If route attributes, such as reachability, are changed, the path along which network traffic passes changes accordingly.

When advertising, receiving, and importing routes, the router implements certain policies based on actual networking requirements to filter routes and change the attributes of the routes. Routing policies serve the following purposes:

- o Control route advertising: Only routes that match the rules specified in a policy are advertised.
- o Control route receiving: Only the required and valid routes are received. This reduces the size of the routing table and improves network security.
- o Filter and control imported routes: A routing protocol may import routes discovered by other routing protocols. Only routes that satisfy certain conditions are imported to meet the requirements of the protocol.
- o Modify attributes of specified routes: Attributes of the routes that are filtered by a routing policy are modified to meet the requirements of the local device.
- o Configure fast reroute (FRR): If a backup next hop and a backup outbound interface are configured for the routes that match a routing policy, IP FRR, VPN FRR, and IP+VPN FRR can be implemented.

Routing policies are implemented using the following procedures:

1. Define rules: Define features of routes to which routing policies are applied. Users define a set of matching rules based on different attributes of routes, such as the destination address and the address of the router that advertises the routes.
2. Implement the rules: Apply the matching rules to routing policies for advertising, receiving, and importing routes.

6. Contributors

The following people have substantially contributed to the definition of the BGP-FS RPD and to the editing of this document:

Peng Zhou
Huawei
Email: Jewpon.zhou@huawei.com

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 would like to thank Acee Lindem, Jeff Haas, Jie Dong, Lucy Yong, Qiandeng Liang, Zhenqiang Li for their comments to this work.

9. IANA Considerations

This document requests assigning a new AFI in the registry "Address Family Numbers" as follows:

| Code Point | Description | Reference |
|-----------------------|--------------------|---------------|
| TBD (36879 suggested) | Routing Policy 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 |
|--------------------|---------------------|---------------|
| TBD(179 suggested) | Routing Policy SAFI | This document |

This document defines a new registry called "Routing Policy NLRI". The allocation policy of this registry is "First Come First Served (FCFS)" according to [RFC8126].

Following code points are defined:

| Code Point | Description | Reference |
|------------|---------------|---------------|
| 1 | Export Policy | This document |
| 2 | Import Policy | This document |

This document requests assigning a code-point from the registry "BGP Community Container Atom Types" as follows:

| TLV Code Point | Description | Reference |
|---------------------|----------------|---------------|
| TBD1 (48 suggested) | RouteAttr Atom | This document |

This document defines a new registry called "Route Attributes Sub-TLV" under RouteAttr Atom TLV. The allocation policy of this registry is "First Come First Served (FCFS)" according to [RFC8126].

Following Sub-TLV code points are defined:

| Code Point | Description | Reference |
|------------|------------------------|---------------|
| 0 | Reserved | |
| 1 | IP Prefix Sub-TLV | This document |
| 2 | AS-Path Sub-TLV | This document |
| 3 | Community Sub-TLV | This document |
| 4 - 255 | To be assigned in FCFS | |

This document defines a new registry called "Attribute Change Sub-TLV" under Parameter(s) TLV. The allocation policy of this registry is "First Come First Served (FCFS)" according to [RFC8126].

Following Sub-TLV code points are defined:

| Code Point | Description | Reference |
|------------|------------------------|---------------|
| 0 | Reserved | |
| 1 | MED Change Sub-TLV | This document |
| 2 | AS-Path Change Sub-TLV | This document |
| 3 - 255 | To be assigned in FCFS | |

10. References

10.1. Normative References

- [I-D.ietf-idr-wide-bgp-communities]
 Raszuk, R., Haas, J., Lange, A., Decraene, B., Amante, S.,
 and P. Jakma, "BGP Community Container Attribute", draft-
 ietf-idr-wide-bgp-communities-05 (work in progress), July
 2018.

- [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>>.
- [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>>.
- [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>>.
- [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>>.

10.2. Informative References

- [I-D.ietf-idr-registered-wide-bgp-communities]
Raszuk, R. and J. Haas, "Registered Wide BGP Community Values", draft-ietf-idr-registered-wide-bgp-communities-02 (work in progress), May 2016.

Authors' Addresses

Zhenbin Li
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Liang Ou
China Telcom Co., Ltd.
109 West Zhongshan Ave, Tianhe District
Guangzhou 510630
China

Email: oul@gsta.com

Yujia Luo
China Telcom Co., Ltd.
109 West Zhongshan Ave, Tianhe District
Guangzhou 510630
China

Email: luoyuj@gsta.com

Sujian Lu
Tencent
Tengyun Building, Tower A , No. 397 Tianlin Road
Shanghai, Xuhui District 200233
China

Email: jasonlu@tencent.com

Huaimo Chen
Futurewei
Boston, MA
USA

Email: Huaimo.chen@futurewei.com

Shunwan Zhuang
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: zhuangshunwan@huawei.com

Haibo Wang
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: rainsword.wang@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 12, 2019

Z. Li
L. Li
Huawei
March 11, 2019

BGP Flow Specification for SRv6
draft-li-idr-flowspec-srv6-00

Abstract

This draft proposes BGP flow specification rules that are used to filter SRv6 packets.

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 September 12, 2019.

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. Definitions and Acronyms | 3 |
| 3. The Flow Specification Encoding for SRv6 | 3 |
| 4. Security Considerations | 5 |
| 5. IANA | 5 |
| 6. Contributors | 6 |
| 7. Acknowledgments | 6 |
| 8. References | 6 |
| Authors' Addresses | 6 |

1. Introduction

BGP Flow Specification (BGP-FS) [RFC5575] 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.

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 Segment Routing Header (SRH) [I-D.ietf-6man-segment-routing-header]. SRv6 Network Programming [I-D.ietf-spring-srv6-network-programming] defined the SRv6 network programming concept and its most basic functions. SRv6 SID will have the form LOC:FUNCT:ARGS::.

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.)

ARGS: A function may require additional arguments that would be placed immediately after the FUNCT

This document specifies a new subset of BGP-FS component types to support Segment Routing over IPv6 data plane (SRv6) filtering.

2. Definitions and Acronyms

- o FS: Flow Specification
- o SR: Segment Routing
- 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

This document proposes new flow specifications rules that is encoded in NLRI. The following new component types are defined

- o Whole SID

Type TBD1 - Whole SID

Encoding: <type (1 octet), length(1 octet), [op, value]+>

Contains a set of {operator, value} pairs that are used to match the SID/binding SID or a range of whole SID.

The operator byte is encoded as:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|----|----|----|---------|---|---|
| + | + | + | + | + | + | + | + |
| e | a | lt | gt | eq | reserve | | |
| + | + | + | + | + | + | + | + |

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 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.

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 SID or a range of SID(e.g. less than SID1 and greater than SID2).

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
+-----+-----+-----+-----+-----+-----+-----+-----+
~                               SID(128bits)                               ~
+-----+-----+-----+-----+-----+-----+-----+-----+

```

The format of SID is described in
[I-D.ietf-6man-segment-routing-header] and
[I-D.filsfils-spring-srv6-network-programming]

o Some bits of SID to match

For some scenarios route policy with the whole 128 bits SID matching is too long and not necessary.
[I-D.filsfils-spring-srv6-network-programming] defined the format of SID is LOC:FUNCT:ARGS::. In some scenarios, traffic packets can just match Locator, Function ID, Argument or combine of these different fields rather than whole 128 bits SID. This document defines a set of new component type TBD2 to reduce the length of matching.

Type TBD2 - Some bits of SID

Encoding: <type (1 octet), length(1 octet), [op, value]+>

Contains a set of {operator, value} pairs that are used to match some bits of SID.

The operator byte is encoded as:

```

      0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| e | a |           type           | reserve |
+---+---+---+---+---+---+---+---+

```

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 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.

type:

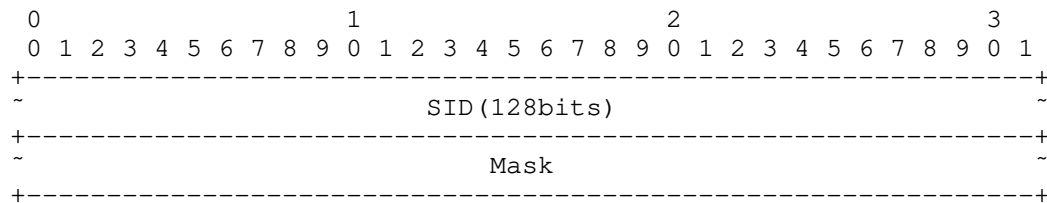
0000 : SID's LOC bits

0001 : SID's FUNCT bits

0010 : SID's LOC:FUNCT bits

0011 : SID's FUNCT:ARGS bits

The value field is encoded as SID with mask to match bits as type defined:



4. Security Considerations

No new security issues are introduced to the BGP protocol by this specification.

5. 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 | SID |
| TBD2 | This Draft | Some bits of SID |

6. Contributors

TBD

7. Acknowledgments

TBD

8. References

- [I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J.,
daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6
Network Programming", draft-filsfils-spring-srv6-network-
programming-07 (work in progress), February 2019.
- [I-D.ietf-6man-segment-routing-header]
Filsfils, C., Previdi, S., Leddy, J., Matsushima, S., and
d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header
(SRH)", draft-ietf-6man-segment-routing-header-16 (work in
progress), February 2019.
- [I-D.ietf-idr-flowspec-l2vpn]
Weiguo, H., Eastlake, D., Uttaro, J., Litkowski, S., and
S. Zhuang, "BGP Dissemination of L2VPN Flow Specification
Rules", draft-ietf-idr-flowspec-l2vpn-09 (work in
progress), January 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>>.
- [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>>.
- [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

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

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.

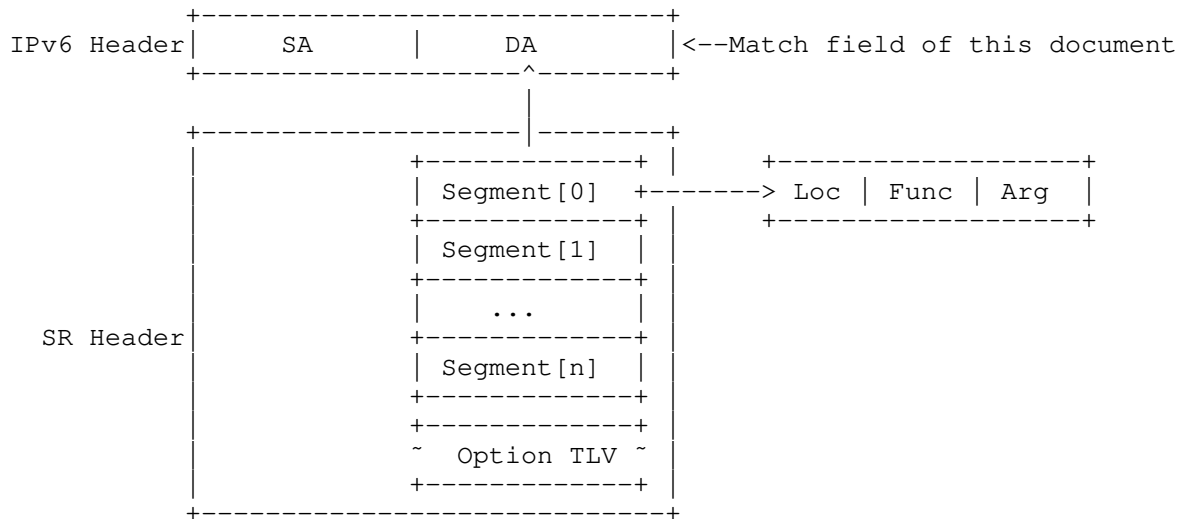


Figure 1: Match Field

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    30  10  40    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

IDR WorkGroup
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2020

M. Zheng
A. Lindem
Cisco Systems
J. Haas
Juniper Networks, Inc.
July 7, 2019

BGP BFD Strict-Mode
draft-merciaz-idr-bgp-bfd-strict-mode-02

Abstract

This document specifies extensions to RFC4271 BGP-4 that enable a BGP speaker to negotiate additional Bidirectional Forwarding Detection (BFD) extensions using a BGP capability. This BFD capability enables a BGP speaker to prevent a BGP session from being established until a BFD session is established. It is referred to as BGP BFD "strict-mode". BGP BFD strict-mode will be supported when both the local speaker and its remote peer are BFD strict-mode capable.

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 January 8, 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. Requirements Language | 3 |
| 3. BFD Capability | 3 |
| 4. Operation | 4 |
| 5. Manageability Considerations | 5 |
| 6. Security Considerations | 5 |
| 7. IANA Considerations | 5 |
| 8. Acknowledgement | 5 |
| 9. Normative References | 6 |
| Authors' Addresses | 6 |

1. Introduction

Bidirectional Forwarding Detection BFD [RFC5882] enables routers to monitor data plane connectivity and to detect faults in the bidirectional forwarding path between them. This capability is leveraged by routing protocols such as BGP [RFC4271] to rapidly react to topology changes in the face of path failures.

The BFD interaction with BGP is specified in Section 10.2 of [RFC5882]. When BFD is enabled for a BGP neighbor, faults in the bidirectional forwarding detected by BFD result in session termination. It is possible in some failure scenarios for the network to be in a state such that a BGP session may be established but a BFD session cannot be established. In some other scenarios, it may be possible to establish a BGP session, but a degraded or poor-quality link may result in the corresponding BFD session going up and down frequently.

To avoid situations which result in routing churn and to minimize the impact of network interruptions, it will be beneficial to disallow BGP to establish a session until BFD session is successfully established and has stabilized. We refer to this mode of operation as BGP BFD "strict-mode". However, always using "strict-mode" would preclude BGP operation in an environment where not all routers support BFD strict-mode or have BFD enabled. This document defines BGP "strict-mode" operation as preventing BGP session establishment until both the local and remote speakers have a stable BFD session. The document also specifies the BGP protocol extensions for BGP capability [RFC5492] for announcing BFD parameters including a BGP

speaker's support for "strict-mode", i.e., requiring a BFD session for BGP session establishment.

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. BFD Capability

The BGP Capability [RFC5492] for BFD parameters will allow a BGP speaker's BFD capabilities including its support for BFD strict-mode. This capability is defined as follows:

Capability code: TBD

Capability length: 1 octet

Capability value: Consists of 1 octet BFD flags as follows:

```
+-----+
| BFD Flags (8 bits) |
+-----+
```

The use and meaning of the fields are as follows:

BFD Flags: This field contains bit flags relating to BFD.

```
  0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
| S | Reserved      |
+---+---+---+---+---+---+
```

The most significant bit is defined as state of Strict-Mode ("Strict-Mode", or "S") bit, which can be used by a BGP speaker to signal its support for BFD Strict-mode. When set (value 1), this bit indicates that the BGP speaker has the BFD "Strict-mode" enabled. If both local BGP speaker and its peer have BFD strict-mode enabled, then BGP session establishment will be prevented until a BFD session is established between the peering addresses. A BGP speaker with BFD

strict-mode enabled MUST advertise the BFD capability with "S" bit set.

The remaining bits are reserved and SHOULD be set to zero by the sender and MUST be ignored by the receiver.

4. Operation

A BGP speaker which supports capabilities advertisement and has BFD strict-mode enabled MUST include the BGP BFD capability with the "S" Bit set in the BGP capabilities it advertises.

A BGP speaker which supports BFD capability, examines the list of capabilities present in the Capabilities BFD Parameter that the speaker receives from its peer. If both the local and remote BGP speakers have BFD strict-mode enabled, the BGP finite state machine does not transition to the Established state from OpenSent or OpenConfirm state [RFC4271] until the BFD session is in the Up state (see below for AdminDown state). This means that a KEEPALIVE message is not sent nor is the KeepaliveTimer set.

If the BFD session does not transition to the Up state, and the HoldTimer has been negotiated to a non-zero value, the BGP FSM will close the session appropriately. If the HoldTimer has been negotiated to a zero value, the session should be closed after a time of X. This time X is referred as "BGP BFD Hold time". The proposed default BGP BFD Hold time value is 30 seconds. The BGP BFD Hold time value is configurable.

If BFD session is in the AdminDown state, then the BGP finite state machine will proceed normally without input from BFD. This means that BFD session "AdminDown" state WILL NOT prevent the BGP state transition to Established state from OpenConfirm.

Once the BFD session has transitioned to the Up state, the BGP FSM may proceed to transition to the Established state from the OpenSent or OpenConfirm state appropriately. I.e. a KEEPALIVE message is sent, and the KeepaliveTimer is started.

If either BGP peer has not advertised the BFD Capability with strict-mode enabled, then a BFD session WILL NOT be required for the BGP session to reach Established state. This does not preclude usage of BFD after BGP session establishment [RFC5882].

5. Manageability Considerations

Auto-configuration is possible for the enabling BGP BFD restrict-mode. However, the configuration automation is out of the scope of this document.

A BGP NOTIFICATION message subcode indicating BFD Hold timer expiration may be required for network management. (To be discussed in the next revision of this document.)

6. Security Considerations

The mechanism defined in this document interacts with the BGP finite state machine when so configured. The security considerations of BFD thus become considerations for BGP-4 [RFC4271] so used. The use of the BFD Authentication mechanism defined in [RFC5880] is thus RECOMMENDED when used to protect BGP-4 [RFC4271].

7. IANA Considerations

This document defines a new BGP capability - BFD Capability. The Capability Code for BFD Capability is TBD.

IANA is requested to establish a "BGP BFD Capability Flags" registry within the "Border Gateway Protocol (BGP) Parameters" grouping. The Registration Procedure should be Standards Action, the initial values as follows:

| Bit Position | Name | Short Name | Reference |
|--------------|-------------|------------|---------------|
| 0 | Strict-Mode | S | this document |
| 1-7 | Unassigned | | this document |

8. Acknowledgement

The authors would like to acknowledge the review and inputs from Shyam Sethuram, Mohammed Mirza, Bruno Decraene, and Carlos Pignataro.

9. 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>>.
- [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>>.
- [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>>.
- [RFC5882] Katz, D. and D. Ward, "Generic Application of Bidirectional Forwarding Detection (BFD)", RFC 5882, DOI 10.17487/RFC5882, June 2010, <<https://www.rfc-editor.org/info/rfc5882>>.
- [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

Mercia Zheng
Cisco Systems
821 Alder Drive
MILPITAS, CALIFORNIA 95035
UNITED STATES

Email: merciarz@cisco.com

Acee Lindem
Cisco Systems
301 Midenhall Way
GARY, NC 27513
UNITED STATES

Email: acee@cisco.com

Jeffrey Haas
Juniper Networks, Inc.
1133 Innovation Way
SUNNYVALE, CALIFORNIA 94089
UNITED STATES

Email: jhaas@juniper.net