Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: November 9, 2014

M. Bhatia
T. Ranganath
Alcatel-Lucent
C. Pignataro
Cisco Systems
S. Aldrin
Huawei Technologies
May 8, 2014

OSPF extensions to advertise S-BFD Target Discriminator
draft-bhatia-ospf-sbfd-discriminator-00

Abstract

This document defines a new OSPF Router Information (RI) TLV that allows OSPF routers to flood the S-BFD discriminator values associated with a target network identifier. This mechanism is applicable to both OSPFv2 and OSPFv3.

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

Copyright Notice

Table of Contents

1. Introduction

Seamless Bidirectional Forwarding Detection (S-BFD), specified in [I-D.akiya-bfd-seamless-base], is a simplified mechanism for using BFD with many negotiations eliminated. This is achieved by using unique network-wide discriminators to identify the Network Targets (e.g., IP addresses). These S-BFD discriminators can be advertised by the IGPs, and this document concerns itself with OSPF. Specifically, this document defines a new TLV (named the S-BFD Discriminator TLV) to be carried within the OSPF Router Information LSA ([RFC4970]).

1.1. Relationship between OSPF and S-BFD

This document, implicitly, defines a relationship between OSPF and S-BFD. S-BFD assigns one or more Discriminators to each S-BFD reflector node. OSPF, in turn, learns about these from S-BFD, and floods them in the newly defined TLV. After this information is flooded, it is stored in all the OSPF nodes such that S-BFD initiators can map out target nodes to target Discriminators, and can therefore construct the S-BFD probe.

2. Implementation

This extension makes use of the Router Information (RI) Opaque LSA, defined in [RFC4970] , for both OSPFv2 [RFC2328] and OSPFv3 [RFC5340], by defining a new OSPF Router Information (RI) TLV: the S-BFD Discriminator TLV.

The S-BFD Discriminator TLV is OPTIONAL. Upon receipt of the TLV, a router may decide to ignore this TLV or install the S-BFD discriminator in BFD Target Identifier Table.

2.1. S-BFD Discriminator TLV

The format of the S-BFD Discriminator TLV is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Discriminator 1                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Discriminator 2 (Optional)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Discriminator n (Optional)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type - S-BFD Discriminator TLV Type

Length - Total length of the discriminator (Value field) in octets, not including the optional padding. The Length is a multiple of 4 octets, and consequently specifies how many Discriminators are included in the TLV.

Value - S-BFD network target discriminator value or values.

Routers that do not recognize the S-BFD Discriminator TLV Type MUST ignore the TLV. S-BFD discriminator is associated with the BFD Target Identifier type, that allows demultiplexing to a specific task or service.

2.2. Flooding Scope

The flooding scope for S-BFD Discriminator information advertised through OSPF can be limited to one or more OSPF areas, or can be extended across the entire OSPF routing domain.

Note that the S-BFD session may be required to pan multiple areas, in which case the flooding scope may comprise these areas. This could be the case for an ABR, for instance, advertising the S-BFD Discriminator information within the backbone area and/or a subset of its attached IGP area(s).

The S-BFD Discriminator TLV is advertised within OSPFv2 Router Information LSAs (Opaque type of 4 and Opaque ID of 0) or OSPFv3 Router Information LSAs (function code of 12), which are defined in [RFC4970]. As such, elements of procedure are inherited from those defined in [RFC4970].

In OSPFv2, the flooding scope is controlled by the opaque LSA type (as defined in [RFC5250]) and in OSPFv3, by the S1/S2 bits (as defined in [RFC5340]). If the flooding scope is area local, then the S-BFD Discriminator TLV MUST be carried within an OSPFv2 type 10 router information LSA or an OSPFV3 Router Information LSA with the S1 bit set and the S2 bit clear. If the flooding scope is the entire IGP domain, then the S-BFD Discriminator TLV MUST be carried within an OSPFv2 type 11 Router Information LSA or OSPFv3 Router Information LSA with the S1 bit clear and the S2 bit set.

When the S-BFD Reflector is deactivated, the OSPF speaker advertising this S-BFD Discriminator MUST originate a new Router Information LSA that no longer includes the corresponding S-BFD Discriminator TLV, provided there are other TLVs in the LSA. If there are no other TLVs in the LSA, it MUST either send an empty Router Information LSA or purge it by prematurely ageing it.

For intra-area reachability, the S-BFD Discriminator TLV information regarding a specific target identifier is only considered current and useable when the router advertising this information is itself reachable via OSPF calculated paths in the same area of the LSA in which the S-BFD Discriminator TLV appears. In the case of domain-wide flooding, i.e., where the originator is sitting in a remote area, the mechanism described in section 5 of [RFC5250] should be used.

A change in information in the S-BFD Discriminator TLV MUST NOT trigger any SPF computation at a receiving router.

3. Backward Compatibility

The S-BFD Discriminator TLV defined in this document does not introduce any interoperability issues.

A router not supporting the S-BFD Discriminator TLV will just silently ignore the TLV as specified in [RFC4970].

4. Security Considerations

This document defines OSPF extensions to distribute the S-BFD discriminator within an administrative domain. Hence the security of the S-BFD discriminator distribution relies on the security of OSPF.

OSPF provides no encryption mechanism for protecting the privacy of LSAs and, in particular, the privacy of the S-BFD discriminator advertisement information. This however is not a concern as there isn’t any need to hide the discriminator value that can be used to reach the Reflectors.

5. IANA Considerations

IANA has defined a registry for TLVs carried in the Router Information LSA defined in [RFC4970]. IANA needs to assign a new TLV codepoint for the S-BFD Discriminator TLV carried within the Router Information LSA.

Value	TLV Name	Reference
TBD	S-BFD Discriminator	(this document)

6. Acknowledgements

The authors would like to thank Nobo Akiya, Les Ginsberg, Mach Chen and Peter Psenak for insightful comments and useful suggestions.

7. References

7.1. Normative References

[I-D.akiya-bfd-seamless-base]
Akiya, N., Pignataro, C., Ward, D., Bhatia, M., and J. Networks, "Seamless Bidirectional Forwarding Detection (S-BFD)", draft-akiya-bfd-seamless-base-03 (work in progress), April 2014.

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC4970] Lindem, A., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 4970, July 2007.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

7.2. Informative References

[RFC5250] Berger, L., Bryskin, I., Zinin, A., and R. Coltun, "The OSPF Opaque LSA Option", RFC 5250, July 2008.

Authors' Addresses

Manav Bhatia
Alcatel-Lucent

Email: manav.bhatia@alcatel-lucent.com

Trilok Ranganath
Alcatel-Lucent

Email: trilok.ranganatha@alcatel-lucent.com

Carlos Pignataro
Cisco Systems

Email: cpignata@cisco.com

Sam Aldrin
Huawei Technologies

Email: aldrin.ietf@gmail.com

Internet Draft
<draft-chen-ospf-transition-to-ospfv3-01.txt>
Category: Informational

I. Chen
A. Lindem
Ericsson
R. Atkinson
Consultant

Expires in 6 months
July 2, 2014

OSPFv3 over IPv4 for IPv6 Transition
<draft-chen-ospf-transition-to-ospfv3-01.txt>

Status of this Memo

Distribution of this memo is unlimited.

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/1id-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 date.

Copyright Notice

Abstract

This document defines a mechanism to use IPv4 to transport OSPFv3 packets, in order to facilitate transition from IPv4-only to IPv6 and dual-stack within a routing domain. Using OSPFv3 over IPv4 with the existing OSPFv3 Address Family extension can simplify transition from an OSFPv2 IPv4-only routing domain to an OSPFv3 dual-stack routing domain.

Table of Contents

1. Introduction

To facilitate transition from IPv4 [RFC791] to IPv6 [RFC2460], dual-stack or IPv6 routing protocols should be gradually deployed. Dual-stack routing protocols, such as Border Gateway Protocol [RFC4271], have an advantage during the transition, because both IPv4 and IPv6 topologies can be transported using either IPv4 or IPv6. Some IPv4-specific and IPv6-specific routing protocols share enough similarities in their protocol packet formats and protocol signaling that it is trivial to deploy an initial IPv6 routing domain by carrying the routing protocol over IPv4 initially, thereby allowing IPv6 routing domains be deployed and tested before decommissioning IPv4 and moving to an IPv6-only network.

In the case of the Open Shortest Path First (OSPF) interior gateway routing protocol (IGP), OSPFv2 [RFC2328] is the IGP deployed over IPv4, while OSPFv3 [RFC5340] is the IGP deployed over IPv6. OSPFv3 further supports multiple address families [RFC5838], including both the IPv6 unicast address family and the IPv4 unicast address family. Consequently, it is possible to deploy OSPFv3 over IPv4 without any changes either to OSPFv3 or to IPv4. During the transition to IPv6, future OSPF extension can focus on OSPFv3 and OSPFv2 can move into maintenance mode.

This document specifies how to use IPv4 packets to transport OSPFv3 packets. The mechanism takes advantage of the fact that OSPFv2 and OSPFv3 share the same IP protocol number, 89. Additionally, the OSPF packet header for both OSPFv2 and OSPFv3 places the OSPF header version (i.e., the field that distinguishes an OSPFv2 packet from an OSPFv3 packet) in the same location.

This document does not attempt to connect an IPv4 topology and an IPv6 topology that are not congruent. In normal operation, it is expected that the IPv4 topology within the OSPF domain will be congruent with the IPv6 topology of that OSPF domain. In such cases, it is expected either that all OSPFv3 packets will be transported

over IPv4 or that all OSPFv3 packets will be transported over IPv6.

If the IPv4 topology and IPv6 topology are not identical, the most likely cause is that some parts of the network deployment have not yet been upgraded to support both IPv4 and IPv6. In situations where the IPv4 deployment is a proper superset of the IPv6 deployment, it is expected that OSPFv3 packets would be transported over IPv4, until the rest of the network deployment is upgraded to support IPv6 in addition to IPv4. In situations where the IPv6 deployment is a proper superset of the IPv4 deployment, it is expected that OSPFv3 would be transported over IPv6.

Throughout this document, OSPF is used when the text applies to both OSPFv2 and OSPFv3. OSPFv2 or OSPFv3 is used when the text is specific to one version of the OSPF protocol. Similarly, IP is used when the text describes either version of the Internet protocol. IPv4 or IPv6 is used when the text is specific to a single version of the protocol.

2. Encapsulation in IPv4

Unlike 6to4 encapsulation [RFC3056] that tunnels IPv6 traffic through an IPv4 network, an OSPFv3 packet can be directly encapsulated within an IPv4 packet as the payload, without the IPv6 packet header, as illustrated in Figure 1. For OSPFv3 transported over IPv4, the IPv4 packet has an IPv4 protocol type of 89, denoting that the payload is an OSPF packet. The payload of the IPv4 packet consists of an OSPFv3 packet, beginning with the OSPF packet header with the OSPF version number set to 3.

An OSPFv3 packet followed by an OSPF link-local signaling (LLS) extension data block [RFC5613] encapsulated in an IPv4 packet is illustrated in Figure 2.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  ^
|   4   |  IHL  |Type of Service|          Total Length         |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|         Identification        |Flags|      Fragment Offset    |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live | Protocol 89   |         Header Checksum       | IPv4
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ Header
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                    Destination Address                        |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                    Options                    |    Padding    |  v
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  ^
|       3       |     Type      |         Packet length         |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Router ID                             | OSPFv3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ Header
|                          Area ID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|          Checksum             |  Instance ID  |      0        |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  v
|                        OSPFv3 Body ...                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: An IPv4 packet encapsulating an OSPFv3 packet.

```
+---------------+
| IPv4 Header   |
+---------------+
| OSPFv3 Header |
|...............|
|               |
| OSPFv3 Body   |
|               |
+---------------+
|               |
| LLS Data      |
|               |
+---------------+
```

Figure 2: The IPv4 packet encapsulating an OSPFv3 packet with a trailing OSPF link-local signaling data block.

2.1. Source Address

For OSPFv3 over IPv4, the source address is the IPv4 interface address for the interface over which the packet is transmitted. All OSPFv3 routers on the link MUST share the same IPv4 subnet for IPv4 transport to function correctly.

2.2. Destination Address

As defined in OSPFv2, the IPv4 destination address of an OSPF protocol packet is either an IPv4 multicast address or the IPv4 unicast address of an OSPFv2 neighbor. Two well-known link-local multicast addresses are assigned to OSPFv2, the AllSPFRouters address (224.0.0.5) and the AllDRouters address (224.0.0.6). The multicast address used depends on the OSPF packet type, the OSPF interface type, and the OSPF router’s role on multi-access networks.

Thus, for an OSPFv3 over IPv4 packet to be sent to AllSPFRouters, the destination address field in the IPv4 packet should be 224.0.0.5. For an OSPFv3 over IPv4 packet to be sent to AllDRouters, the destination address field in the IPv4 packet should be 224.0.0.6.

When an OSPF router sends a unicast OSPF packet over a connected interface, the destination of such an IP packet is the address assigned to the receiving interface. Thus, a unicast OSPFv3 packet transported in an IPv4 packet would specify the OSPFv3 neighbor’s IPv4 address as the destination address.

2.3. Operation over Virtual Link

When an OSPF router sends an OSPF packet over a virtual link, the receiving router is a router which is not directly connected to the sending router. Thus, the destination IP address of the IP packet must be a reachable unicast IP address of the receiving router. Because IPv6 is the presumed Internet protocol and an IPv4 destination is not routable, the OSPFv3 address family extension [RFC5838] specifies that only IPv6 address family virtual links are supported.

As illustrated in Figure 1, this document specifies OSPFv3 transport over IPv4. As a result, an IPv4 packet in which the destination field is a unicast IPv4 address assigned to the virtual router is routable, and OSPFv3 virtual links in IPv4 unicast address families can be supported. Hence, the restriction in Section 2.8 of RFC 5838 [RFC5838] is removed. If IPv4 transport, as specified herein, is used for IPv6 address families, virtual

links cannot be supported. Hence, it is RECOMMENDED to use the IP transport matching the address family in OSPF routing domains requiring virtual links.

3. IPv4-only Use Case

OSPFv3 only requires IPv6 link-local addresses to establish a routing domain, and does not require IPv6 global-scope addresses to establish a routing domain. However, IPv6 over Ethernet [RFC2464] uses a different EtherType (0x86dd) from IPv4 (0x0800) and also from the Address Resolution Protocol (ARP) (0x0806) [RFC826] that is used with IPv4.

Some existing deployed link-layer equipment only supports IPv4 and ARP. Such equipment contains hardware filters keyed on the EtherType field of the Ethernet frame to filter which frames will be accepted into that link-layer equipment. Because IPv6 uses a different EtherType, IPv6 framing for OSPFv3 won’t work with that equipment. In other cases, PPP might be used over a serial interface, but again only IPv4 over PPP might be supported over that interface. It is hoped that equipment with such limitations will be replaced eventually.

In some locations, especially locations with less communications infrastructure, satellite communications (SATCOM) is used to reduce deployment costs for data networking. SATCOM often has lower cost to deploy than running new copper or optical cables for long distances to connect remote areas. Also, in a wide range of locations including places with good communications infrastructure, Very Small Aperture Terminals (VSAT) often are used by banks and retailers to connect their stores to their main offices.

Some widely deployed VSAT equipment has either (A) Ethernet interfaces that only support Ethernet Address Resolution Protocol (ARP) and IPv4, or (B) serial interfaces that only support IPv4 and Point-to-Point Protocol (PPP) packets. Such deployments and equipment still can deploy and use OSPFv3 over IPv4 today, and then later migrate to OSPFv3 over IPv6 after equipment is upgraded or replaced. This can have lower operational costs than running OSPFv2 and then trying to make a flag-day switch to running OSPFv3. By running OSPFv3 over IPv4 now, the eventual transition to dual-stack, and then to IPv6-only can be optimized.

4. Security Considerations

As described in [RFC4552], OSPFv3 uses IPsec [RFC4301] for authentication and confidentiality. Consequently, an OSPFv3 packet transported within an IPv4 packet requires IPsec to provide

authentication and confidentiality. Further work such as [ipsecospf] would be required to support IPsec protection for OSPFv3 over IPv4 transport.

An optional OSPFv3 Authentication Trailer [RFC6506] also has been defined as an alternative to using IPsec. The calculation of the authentication data in the Authentication Trailer includes the source IPv6 address to protect an OSPFv3 router from Man-in-the-Middle attacks. For IPv4 encapsulation as described herein, the IPv4 source address should be placed in the first 4 octets of Apad followed by the hexadecimal value 0x878FE1F3 repeated (L-4)/4 times, where L is the length of hash measured in octet.

The processing of the optional Authentication Trailer is contained entirely within the OSPFv3 protocol. In other words, each OSPFv3 router instance is responsible for the authentication, without involvement from IPsec or any other IP layer function. Consequently, except for calculation of the value Apad, transporting OSPFv3 packets using IPv4 does not change the operation of the optional OSPFv3 Authentication Trailer.

5. IANA Considerations

No actions are required from IANA as result of the publication of this document.

6. References

6.1. Normative References

[RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

[RFC2328] Moy, J., "OSPF Version 2", STD54, RFC 2328, April 1998.

[RFC5838] Lindem, A., Ed., Mirtorabi, S., Roy, A., Barnes, M., and R. Aggarwal, "Support of Address Families in OSPFv3", RFC 5838, April 2010.

6.2. Informative References

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A

Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[RFC3056] Carpenter, B. and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds", RFC 3056, February 2001.

[RFC5613] Zinin, A., Roy, A., Nguyen, L., Friedman, B., and D. Yeung, "OSPF Link-Local Signaling", RFC 5613, August 2009.

[RFC826] Plummer, D., "Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, November 1982.

[RFC2464] Crawford, M., "Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, December 1998.

[RFC4552] Gupta, M. and N. Melam, "Authentication/Confidentiality for OSPFv3", RFC 4552, June 2006.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC6506] Bhatia, M., Manral, V., and A. Lindem, "Supporting Authentication Trailer for OSPFv3", RFC 6506, February 2012.

[ipsecospf] Gupta, M. and Melam, M, Work in progress, "draft-gupta-ospf-ospfv2-sec-01.txt", August 2009.

Authors' Addresses

I. Chen
Ericsson
Email: ing-wher.chen@ericsson.com

A. Lindem
Ericsson
Email: acee.lindem@ericsson.com

R. Atkinson
Consultant

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: April 29, 2015

H. Chen
R. Li
A. Kumar S N
Huawei Technologies
G. Cauchie

A. Retana
Cisco Systems, Inc.
N. So
Tata Communications
V. Liu
China Mobile
M. Toy
Comcast
L. Liu
UC Davis

October 26, 2014

OSPF Topology-Transparent Zone
draft-chen-ospf-ttz-09.txt

Abstract

This document presents a topology-transparent zone in a domain. A topology-transparent zone comprises a group of routers and a number of links connecting these routers. Any router outside of the zone is not aware of the zone. The information about the links and routers inside the zone is not distributed to any router outside of the zone. Any link state change such as a link down inside the zone is not seen by any router outside of the zone.

Status of this Memo

Table of Contents

1. Introduction

The number of routers in a network becomes larger and larger as the Internet traffic keeps growing. Through splitting the network into multiple areas, we can extend the network further. However, there are a number of issues when a network is split further into more areas.

At first, dividing a network from one area into multiple areas or from a number of existing areas to even more areas is a very challenging and time consuming task since it is involved in significant network architecture changes. Considering the one area case, originally the network has only one area, which is the backbone. This original backbone area will be split into a new backbone and a number of non backbone areas. In general, each of the non backbone areas is connected to the new backbone area through the area border routers between the non backbone and the backbone area. There is not any direct connection between any two non backbone areas. Each area border router summarizes the topology of its attached non backbone area for transmission on the backbone area, and hence to all other area border routers.

Secondly, the services carried by the network may be interrupted while the network is being split from one area into multiple areas or from a number of existing areas into even more areas.

Furthermore, it is complex for a Multi-Protocol Label Switching (MPLS) Traffic Engineering (TE) Label Switching Path (LSP) crossing multiple areas to be setup. In one option, a TE path crossing multiple areas is computed by using collaborating Path Computation Elements (PCEs) [RFC5441] through the PCE Communication Protocol (PCEP)[RFC5440], which is not easy to configure by operators since the manual configuration of the sequence of domains is required. Although this issue can be addressed by using the Hierarchical PCE, this solution may further increase the complexity of network design. Especially, the current PCE standard method may not guarantee that the path found is optimal.

This document presents a topology-transparent zone in an area and describes extensions to OSPF for supporting the topology-transparent zone, which is scalable and resolves the issues above.

A topology-transparent zone comprises a group of routers and a number of links connecting these routers. Any router outside of the zone is not aware of the zone. The information about the links and routers inside the zone is not distributed to any router outside of the zone. Any link state change such as a link down inside the zone is not seen by any router outside of the zone.

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.

3. Requirements

Topology-Transparent Zone (TTZ) may be deployed for resolving some critical issues in existing networks and future networks. The requirements for TTZ are listed as follows:

- o TTZ MUST be backward compatible. When a TTZ is deployed on a set of routers in a network, the routers outside of the TTZ in the network do not need to know or support TTZ.

- o TTZ MUST support at least one more levels of network hierarchies, in addition to the hierarchies supported by existing routing protocols.

- o Users SHOULD be able to easily set up an end to end service crossing TTZs.

- o The configuration for a TTZ in a network SHOULD be minimum.

- o The changes on the existing protocols for supporting TTZ SHOULD be minimum.

4. Topology-Transparent Zone

4.1. Overview of Topology-Transparent Zone

A Topology-Transparent Zone (TTZ) is identified by an Identifier (ID), and it includes a group of routers and a number of links connecting the routers. A TTZ is in an OSPF area.

The ID of a TTZ or TTZ ID is a number that is unique for identifying an entity such as a node in an OSPF domain. It is not zero in general.

In addition to having the functions of an OSPF area, an OSPF TTZ makes some improvements on an OSPF area, which include:

- o An OSPF TTZ is virtualized as the TTZ edge routers connected.

o An OSPF TTZ receives the link state information about the topology outside of the TTZ, stores the information in the TTZ and floods the information through the TTZ to the routers outside of the TTZ.

4.2. An Example of TTZ

The figure below shows an area containing a TTZ: TTZ 600.

```
                TTZ 600
                  \
                   \ ^~^~^~^~^~^~^~^~^~^~^~^~^~
                    (                        )
    ===[R15]========(==[R61]------------[R63]==)======[R29]===
        ||         (   |    \          /    |   )       ||
        ||         (   |     \        /     |   )       ||
        ||         (   |      \      /      |   )       ||
        ||         (   |    ___\    /       |   )       ||
        ||         (   |   /   [R71]        |   )       ||
        ||         (   | [R73] /    \       |   )       ||
        ||         (   |      /      \      |   )       ||
        ||         (   |     /        \     |   )       ||
        ||         (   |    /          \    |   )       ||
    ===[R17]========(==[R65]------------[R67]==)======[R31]===
         \\          (//                    \\)       //
          ||         //v~v~v~v~v~v~v~v~v~v~v~\\      ||
          ||        //                        \\     ||
          ||       //                          \\    ||
           \\     //                            \\  //
       ======[R23]==============================[R25]=====
             //                                     \\
            //                                       \\
```

Figure 1: An Example of TTZ

The area comprises routers R15, R17, R23, R25, R29 and R31. It also contains TTZ 600, which comprises routers R61, R63, R65, R67, R71 and R73, and the links connecting them.

There are two types of routers in a TTZ: TTZ internal routers and TTZ edge routers. A TTZ internal router is a router inside the TTZ and its adjacent routers are in the TTZ. A TTZ edge router is a router inside the TTZ and has at least one adjacent router that is outside of the TTZ.

The TTZ in the figure above comprises four TTZ edge routers R61, R63, R65 and R67. Each TTZ edge router is connected to at least one router outside of the TTZ. For instance, router R61 is a TTZ edge

router since it is connected to router R15, which is outside of the TTZ.

In addition, the TTZ comprises two TTZ internal routers R71 and R73. A TTZ internal router is not connected to any router outside of the TTZ. For instance, router R71 is a TTZ internal router since it is not connected to any router outside of the TTZ. It is just connected to routers R61, R63, R65, R67 and R73 in the TTZ.

A TTZ MUST hide the information inside the TTZ from the outside. It MUST NOT directly distribute any internal information about the TTZ to a router outside of the TTZ.

For instance, the TTZ in the figure above MUST NOT send the information about TTZ internal router R71 to any router outside of the TTZ in the routing domain; it MUST NOT send the information about the link between TTZ router R61 and R65 to any router outside of the TTZ.

In order to create a TTZ, we MUST configure the same TTZ ID on the edge routers and identify the TTZ internal links on them. In addition, we SHOULD configure the TTZ ID on every TTZ internal router which indicates that every link of the router is a TTZ internal link.

From a router outside of the TTZ, a TTZ is seen as a group of routers fully connected. For instance, router R15 in the figure above, which is outside of TTZ 600, sees TTZ 600 as a group of TTZ edge routers: R61, R63, R65 and R67. These four TTZ edge routers are fully connected.

In addition, a router outside of the TTZ sees TTZ edge routers having normal connections to the routers outside of the TTZ. For example, router R15 sees four TTZ edge routers R61, R63, R65 and R67, which have the normal connections to R15, R29, R17 and R23, R25 and R31 respectively.

5. Extensions to OSPF Protocols

5.1. Opaque LSAs for TTZ

The link state information about a TTZ includes router LSAs and network LSAs describing the TTZ topology. These LSAs can be contained and distributed in opaque LSAs within the TTZ. Some control information on a TTZ can also be contained and distributed in opaque LSAs within the TTZ. These opaque LSAs are called TTZ opaque LSAs or TTZ LSAs for short.

The following is a general form of a TTZ LSA. It has an LS type = 10 and TTZ-LSA-Type, and contains a number of TLVs.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            LS age             |     Options   | LS Type = 10  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  TTZ-LSA-type |                     Opaque ID                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Advertising Router                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      LS Sequence Number                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         LS checksum           |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                              TLVs                             ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Where TTZ-LSA-type may be TBD1 (TTZ-RT-LSA-type) for TTZ Router LSA, TBD2 (TTZ-NW-LSA-type) for TTZ Network LSA, and TBD3 (TTZ-CT-LSA-type) for TTZ Control LSA.

There are four types of TLVs: TTZ ID TLV, TTZ Router TLV, TTZ network TLV and TTZ Options TLV. A TTZ ID TLV has the following format. It contains a TTZ ID.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       TTZ-ID-TLV-type         |          Length = 4           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            TTZ ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of a TTZ Router TLV is as follows. It contains the contents of a normal router LSA. A TTZ router LSA includes a TTZ ID TLV and a TTZ Router TLV.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       TTZ-RT-TLV-type         |          TLV-Length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|G|   0   |V|E|B|        0      |            # links            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Link ID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Link Data                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     # TOS     |            metric             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                               ...                             ~
```

Where G = 1/0 indicates that the router is an edge/internal router of TTZ. For a router link, the existing eight bit Type field for a router link may be split into two fields as follows:

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| I |         Type-1            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
I bit flag:
  1: Router link is an internal link to a router inside TTZ.
  0: This indicates that the router link is an external link.
Type-1: The kind of the link.
```

For a link inside a TTZ, I bit flag is set to one, indicating that this link is an internal TTZ link. For a link connecting to a router outside of a TTZ from a TTZ edge router, I bit flag is set to zero, indicating that this link is an external TTZ link.

The value of Type-1 may be 1, 2, 3, or 4, which indicates that the kind of a link being described is a point-to-point connection to another router, a connection to a transit network, a connection to a stub network, or a virtual link respectively.

A TTZ Network TLV has the following format. It contains the contents of a normal network LSA. A TTZ network LSA includes a TTZ ID TLV and a TTZ network TLV.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       TTZ-NW-TLV-type         |          TLV-Length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Network Id                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Network Mask                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Attached Router                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                               ...                             ~
```

Where Network ID is the interface address of the DR, which is followed by the contents of a network LSA.

The format of TTZ Options TLV is as follows. A TTZ control LSA contains a TTZ ID TLV and a TTZ Options TLV.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       TTZ-OP-TLV-type         |          Length = 4           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|T|M|N|R|    0                                                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

T = 1: Distributing TTZ Topology Information for Migration
M = 1: Migrating to TTZ
N = 1: Distributing Normal Topology Information for Rollback
R = 1: Rolling back from TTZ

5.2. A TTZ Capability TLV in Router Information LSA

A new bit such as bit 6 for TTZ capability may be defined in the Router Informational Capabilities TLV as follows:

Bit	Capabilities
0	OSPF graceful restart capable [GRACE]
:	...
5	OSPF Experimental TE [EXP-TE]
6	OSPF TTZ capable [OSPF-TTZ]
7-31	Unassigned (Standards Action)

When the OSPF TTZ capable bit is set to one, a TTZ capability TLV must follow the Router Informational Capabilities TLV to indicate a link/router’s TTZ capability and the TTZ to which the link/router

belongs. 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     TTZ-CAP-TLV-Type = 2      |          Length = 8           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            TTZ ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|M|                             0                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

It contains a TTZ ID and a number of TTZ bits. The following bits in the TLV are assigned:

Bit	Meaning
M	Have Migrated to TTZ (i.e., works as TTZ)
1-31	Unassigned (Standards Action)

A link scope RI LSA with a OSPF TTZ capable bit set to one and a TTZ Capability TLV will be used to discover a TTZ neighbor.

6. Constructing LSAs for TTZ

There are three types of LSAs for representing a TTZ: TTZ router LSA, TTZ network LSA and Router LSA for virtualizing TTZ. The first two may be generated by a TTZ router, and the third by a TTZ edge router.

A TTZ router LSA generated by a TTZ router has a TTZ ID TLV and a TTZ Router TLV. The former includes the ID of the TTZ to which the router belongs. The latter contains the links to the router.

A TTZ network LSA for a broadcast link is generated by the DR for the link. It contains a TTZ ID TLV and a TTZ network TLV. The former has the ID of the TTZ to which the link belongs. The latter includes the DR’s address, the network mask, and the routers attached.

A router LSA for virtualizing a TTZ generated by an edge router of the TTZ comprises three groups of links in general.

The first group are the router links connecting the routers outside of the TTZ. These router links are normal router links. There is a router link for every adjacency between this TTZ edge router and a router outside of the TTZ.

The second group are the "virtual" router links. For each of the other TTZ edge routers, there is a point-to-point router link to it.

The cost of the link may be the cost of the shortest path from this TTZ edge router to it within the TTZ.

In addition, the LSA may contain a third group of links, which are stub links for other destinations inside the TTZ. They may be the loopback addresses to be accessed by a node outside of the TTZ.

7. Establishing Adjacencies

This section describes the adjacencies in some different cases.

7.1. Discover TTZ Neighbor over Normal Adjacency

For two routers A and B connected by a P2P link and having a normal adjacency, they discover TTZ each other through a link scope RI LSA with an OSPF TTZ capable bit and a TTZ ID. We call this LSA D-LSA for short. If two ends of the link have the same TTZ ID, A and B are TTZ neighbors. The following is a sequence of events related to TTZ.

```
           A                                  B
    Configure TTZ                            Configure TTZ
                    D-LSA (TTZ-ID=100)
                 ---------------------------> Same TTZ ID
                                             A is B's TTZ Neighbor
                    D-LSA (TTZ-ID=100)
    Same TTZ ID  <---------------------------
    B is A's TTZ Neighbor
```

A sends B a D-LSA with TTZ-ID after the TTZ is configured on it. B sends A a D-LSA with TTZ-ID after the TTZ is configured on it. When A receives the D-LSA from B and determines they have the same TTZ ID, B is A's TTZ neighbor. When B receives the D-LSA from A and determines they have the same TTZ ID, A is B's TTZ neighbor.

For a number of routers connected through a broadcast link and having normal adjacencies among them, they also discover TTZ each other through D-LSAs. The DR for the link "forms" TTZ adjacency with each of the other routers if all the routers attached to the link have the same TTZ ID configured on the connections to the link.

7.2. Establishing TTZ Adjacencies

When a router (say A) is connected via a P2P link to another router (say B) and there is not any adjacency between them over the link, a user configures TTZ on two ends of the link to form a TTZ adjacency.

While A and B are forming an adjacency, they start to discover TTZ each other through D-LSAs in the same way as described above after the normal adjacency is greater than ExStart. When the normal adjacency is full and B becomes A's TTZ neighbor, A forms a TTZ adjacency with B. Similarly, B forms a TTZ adjacency with A.

For a number of routers connected through a broadcast link and having no adjacency among them, they start to form TTZ adjacencies after TTZ is configured on the link. While forming adjacencies, they discover TTZ each other through D-LSAs in the same way as described above after the normal adjacency is greater than ExStart. The DR for the link forms TTZ adjacency with each of the other routers if all the routers attached to the link have the same TTZ ID configured on the connections to the link. Otherwise, the DR does not form any adjacency with any router attached to the link.

An alternative way for forming an adjacency between two routers in a TTZ is to extend hello protocol. Hello protocol is extended to include TTZ ID in LLS of a hello packet. The procedure for handling hellos is changed to consider TTZ ID. If two routers have the same TTZ IDs in their hellos, an adjacency between these two routers is to be formed; otherwise, no adjacency is formed.

7.3. Adjacency between TTZ Edge and Router outside

For an edge router in a TTZ, it forms an adjacency with any router outside of the TTZ that has a connection with it.

When the edge router synchronizes its link state database with the router outside of the TTZ, it sends the router outside of the TTZ the information about all the LSAs except for the LSAs belonging to the TTZ that are hidden from any router outside of the TTZ.

At the end of the link state database synchronization, the edge router originates its own router LSA for virtualizing the TTZ and sends this LSA to the router outside of the TTZ.

From the point of view of the router outside of the TTZ, it sees the other end as a normal router and forms the adjacency in the same way as a normal router. It is not aware of anything about its neighboring TTZ. From the LSAs related to the TTZ edge router in the other end, it knows that the TTZ edge router is connected to each of the other TTZ edge routers and some routers outside of the TTZ.

8. Distribution of LSAs

LSAs can be divided into a couple of classes according to their

distributions. The first class of LSAs is distributed within a TTZ. The second is distributed through a TTZ.

8.1. Distribution of LSAs within TTZ

Any LSA about a link state in a TTZ is distributed within the TTZ. It is not distributed to any router outside of the TTZ. For example, a router LSA generated for a router in a TTZ is distributed within the TTZ and not distributed to any router outside of the TTZ.

Any network LSA generated for a broadcast or NBMA network in a TTZ is distributed in the TTZ and not sent to a router outside of the TTZ.

Any opaque LSA generated for a TTZ internal TE link is distributed within the TTZ and not distributed to any router outside of the TTZ.

8.2. Distribution of LSAs through TTZ

Any LSA about a link state outside of a TTZ received by an edge router of the TTZ is distributed through the TTZ. For example, when an edge router of a TTZ receives an LSA from a router outside of the TTZ, it floods it to its neighboring routers both inside the TTZ and outside of the TTZ. This LSA may be any LSA such as a router LSA that is distributed in a domain.

The routers in the TTZ continue to flood the LSA. When another edge router of the TTZ receives the LSA, it floods the LSA to its neighboring routers both outside of the TTZ and inside the TTZ.

9. Computation of Routing Table

The computation of the routing table on a router is the same as that described in RFC 2328, with one exception. A router in a TTZ MUST ignore the router LSAs generated by the edge routers of the TTZ for virtualizing the TTZ. It computes routes through using the TTZ topology represented by TTZ LSAs and the topology outside of the TTZ.

10. Operations

10.1. Configuring TTZ

This section proposes some options for configuring a TTZ.

1. Configuring TTZ on Every Link in TTZ

If every link in a TTZ is configured with a same TTZ ID as a TTZ

link, the TTZ is determined. A router with some TTZ links and some normal links is a TTZ edge router. A router with only TTZ links is a TTZ internal router.

2. Configuring TTZ on Every Router in TTZ

We may configure a same TTZ ID on every router in the TTZ, and on every edge router’s links connecting to the routers in the TTZ.

A router configured with the TTZ ID on some of its links is a TTZ edge router. A router configured with the TTZ ID only is a TTZ internal router. All the links on a TTZ internal router are TTZ links. This option is simpler than the above one.

10.2. Smooth Migration to TTZ

For a group of routers and a number of links connecting the routers in an area, making them transfer to work as a TTZ without any service interruption may take a few of steps or stages.

At first, users configure the TTZ feature on every router in the TTZ. In this stage, a router does not originate its TTZ router LSA or TTZ network LSAs. It will discover its TTZ neighbors.

Secondly, after configuring the TTZ, users may issue a CLI command on one router in the TTZ, which triggers every router in the TTZ to generate and distribute TTZ information among the routers in the TTZ. When the router receives the command, it originates its TTZ router LSA and TTZ network LSAs as needed, and distributes them to its TTZ neighbors. It also originates a TTZ control LSA with T=1 (indicating TTZ information generation and distribution for migration). When a router in the TTZ receives the LSA with T=1, it originates its TTZ router LSA and TTZ network LSAs as needed. In this stage, every router in the TTZ has dual roles. One is to function as a normal router. The other is to generate and distribute TTZ information.

Thirdly, users SHOULD check whether every router in the TTZ is ready for transferring to work as a TTZ router. A router in the TTZ is ready after it has received all the necessary information from all the routers in the TTZ. This information may be displayed on a router through a CLI command.

And then users may activate the TTZ through using a CLI command such as migrate to TTZ on one router in the TTZ. The router transfers to work as a TTZ router, generates and distributes a TTZ control LSA with M=1 (indicating Migrating to TTZ) after it receives the command.

After a router in the TTZ receives the TTZ control LSA with M=1, it

also transfers to work as a TTZ router. Thus, activating the TTZ on one TTZ router makes every router in the TTZ transfer to work as a TTZ router, which flushes its normal router LSA and network LSAs, computes routes through using the TTZ topology represented by TTZ LSAs and the topology outside of the TTZ.

For an edge router of the TTZ, transferring to work as a TTZ router comprises generating a router LSA to virtualize the TTZ and flooding this LSA to all its neighboring routers.

10.3. Adding a Router into TTZ

When a non TTZ router (say R1) is connected via a P2P link to a TTZ router (say T1) working as TTZ and there is a normal adjacency between them over the link, a user can configure TTZ on two ends of the link to add R1 into the TTZ to which T1 belongs. They discover TTZ each other in the same way as described in section 7.1.

When a number of non TTZ routers are connected via a broadcast link to a TTZ router (say T1) working as TTZ and there are normal adjacencies among them, a user configures TTZ on the connection to the link on every router to add the non TTZ routers into the TTZ to which T1 belongs. The DR for the link "forms" TTZ adjacency with each of the other routers if all the routers have the same TTZ ID configured on the connections to the link.

When a router (say R1) is connected via a P2P link to a TTZ router (say T1) and there is not any adjacency between them over the link, a user can configure TTZ on two ends of the link to add R1 into the TTZ to which T1 belongs. R1 and T1 will form an adjacency in the same way as described in section 7.2.

When a router (say R1) is connected via a broadcast link to a group of TTZ routers on the link and there is not any adjacency between R1 and any over the link, a user can configure TTZ on the connection to the link on R1 to add R1 into the TTZ to which the TTZ routers belong. R1 starts to form an adjacency with the DR for the link after the configuration.

11. Prototype Implementation

11.1. What are Implemented and Tested

1. CLI Commands for TTZ

The CLIs implemented and tested include:

- o the CLIs of the simpler option for configuring TTZ, and

- o the CLIs for controlling migration to TTZ.

2. Extensions to OSPF Protocols for TTZ

All the extensions defined in section "Extensions to OSPF Protocols" are implemented and tested except for rolling back from TTZ. The testing results illustrate:

- o A TTZ is virtualized to outside as its edge routers fully connected. Any router outside of the TTZ sees the edge routers (as normal routers) connecting each other and to some other routers.

- o The link state information about the routers and links inside the TTZ is contained within the TTZ. It is not distributed to any router outside of the TTZ.

- o TTZ is transparent. From a router inside a TTZ, it sees the topology (link state) outside of the TTZ. From a router outside of the TTZ, it sees the topology beyond the TTZ. The link state information outside of the TTZ is distributed through the TTZ.

- o TTZ is backward compatible. Any router outside of a TTZ does not need to support or know TTZ.

3. Smooth Migration to TTZ

The procedures and related protocol extensions for smooth migration to TTZ are implemented and tested. The testing results show:

- o A part of an area is smoothly migrated to a TTZ without any routing disruptions. The routes on every router are stable while the part of the area is being migrated to the TTZ.

- o Migration to TTZ is very easy to operate.

4. Add a Router to TTZ

Adding a router into TTZ is implemented and tested. The testing results illustrate:

- o A router can be easily added into a TTZ and becomes a TTZ router.

- o The router added into the TTZ is not seen on any router outside of the TTZ, but it is a part of the TTZ.

5. Leak TTZ Loopbacks Outside

Leaking loopback addresses in a TTZ to routers outside of the TTZ is implemented and tested. The testing results illustrate:

- o The loopback addresses inside the TTZ are distributed to the routers outside of the TTZ.

- o The loopback addresses are accessable from a router outside of the TTZ.

11.2. Implementation Experience

The implementation of TTZ is relatively easy compared to other features of OSPF. Re-using the existing OSPF code along with additional simple logic does the work. A couple of engineers started to work on implementing the TTZ from the middle of June, 2014 and finished coding it just before IETF 90. After some testing and bug fixes, it works as expected.

In our implementation, the link state information in a TTZ opaque LSA is stored in the same link state database as the link state information in a normal LSA. For each TTZ link in the TTZ opaque LSA stored, there is an additional flag, which is used to differentiate between a TTZ link and a Normal link.

Before migration to TTZ, every router in the TTZ computes its routing table using the normal links. After migration to TTZ, every router in the TTZ computes its routing table using the TTZ links and normal links. In the case that there are one TTZ link and one normal link to select, the TTZ link is used. In SPF calculation, the back-link check passes if and only if the corresponding new additional bit matches. If link type bit is TTZ link, then the lookup is for corresponding TTZ LSA. In case of normal link, the lookup is based on normal link.

12. Security Considerations

The mechanism described in this document does not raise any new security issues for the OSPF protocols.

13. IANA Considerations

TBD

14. Contributors

Veerendranatha Reddy Vallem
Huawei Technologies
Banglore
India
Email: veerendranatharv@huawei.com

15. Acknowledgement

The author would like to thank Acee Lindem, Abhay Roy, Dean Cheng, Russ White, William McCall, Tony Przygienda, Lin Han and Yang Yu for their valuable comments on this draft.

16. References

16.1. Normative References

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC4970] Lindem, A., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 4970, July 2007.

[RFC2370] Coltun, R., "The OSPF Opaque LSA Option", RFC 2370, July 1998.

[RFC5613] Zinin, A., Roy, A., Nguyen, L., Friedman, B., and D. Yeung, "OSPF Link-Local Signaling", RFC 5613, August 2009.

16.2. Informative References

[RFC5441] Vasseur, JP., Zhang, R., Bitar, N., and JL. Le Roux, "A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths", RFC 5441, April 2009.

[RFC5440] Vasseur, JP. and JL. Le Roux, "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, March 2009.

Authors' Addresses

Huaimo Chen
Huawei Technologies
Boston, MA
USA

Email: huaimo.chen@huawei.com

Renwei Li
Huawei Technologies
2330 Central expressway
Santa Clara, CA
USA

Email: renwei.li@huawei.com

Anil Kumar S N
Huawei Technologies
Banglore
India

Email: anil.sn@huawei.com

Gregory Cauchie
FRANCE

Email: greg.cauchie@gmail.com

Alvaro Retana
Cisco Systems, Inc.
7025 Kit Creek Rd.
Raleigh, NC 27709
USA

Email: aretana@cisco.com

Ning So
Tata Communications
2613 Fairbourne Cir.
Plano, TX 75082
USA

Email: ningso01@gmail.com

Vic Liu
China Mobile
No.32 Xuanwumen West Street, Xicheng District
Beijing, 100053
China

Email: liuzhiheng@chinamobile.com

Mehmet Toy
Comcast
1800 Bishops Gate Blvd.
Mount Laurel, NJ 08054
USA

Email: mehmet_toy@cable.comcast.com

Lei Liu
UC Davis
CA
USA

Email: liulei.kddi@gmail.com

Network Working Group
Internet-Draft
Updates: 5340, 5838 (if approved)
Intended status: Standards Track
Expires: July 29, 2018

A. Lindem
A. Roy
Cisco Systems
D. Goethals
Nokia
V. Reddy Vallem

F. Baker
January 25, 2018

OSPFv3 LSA Extendibility
draft-ietf-ospf-ospfv3-lsa-extend-23.txt

Abstract

OSPFv3 requires functional extension beyond what can readily be done with the fixed-format Link State Advertisement (LSA) as described in RFC 5340. Without LSA extension, attributes associated with OSPFv3 links and advertised IPv6 prefixes must be advertised in separate LSAs and correlated to the fixed-format LSAs. This document extends the LSA format by encoding the existing OSPFv3 LSA information in Type-Length-Value (TLV) tuples and allowing advertisement of additional information with additional TLVs. Backward compatibility mechanisms are also described.

This document updates RFC 5340, "OSPF for IPv6", and RFC 5838, "Support of Address Families in OSPFv3" by providing TLV-based encodings for the base OSPFv3 unicast support and OSPFv3 address family support.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

OSPFv3 requires functional extension beyond what can readily be done with the fixed-format Link State Advertisement (LSA) as described in RFC 5340 [OSPFV3]. Without LSA extension, attributes associated with OSPFv3 links and advertised IPv6 prefixes must be advertised in separate LSAs and correlated to the fixed-format LSAs. This document extends the LSA format by encoding the existing OSPFv3 LSA information in Type-Length-Value (TLV) tuples and allowing advertisement of additional information with additional TLVs. Backward compatibility mechanisms are also described.

This document updates RFC 5340, "OSPF for IPv6", and RFC 5838, "Support of Address Families in OSPFv3" by providing TLV-based encodings for the base OSPFv3 support [OSPFV3] and OSPFv3 address family support [OSPFV3-AF].

A similar extension was previously proposed in support of multi-topology routing. Additional requirements for OSPFv3 LSA extension include source/destination routing, route tagging, and others.

A final requirement is to limit the changes to OSPFv3 to those necessary for TLV-based LSAs. For the most part, the semantics of existing OSPFv3 LSAs are retained for their TLV-based successor LSAs described herein. Additionally, encoding details, e.g., the representation of IPv6 prefixes as described in section A.4.1 in RFC 5340 [OSPFV3], have been retained. This requirement was included to increase the expedience of IETF adoption and deployment.

The following aspects of OSPFv3 LSA extension are described:

1. Extended LSA Types

2. Extended LSA TLVs

3. Extended LSA Formats

4. Backward Compatibility

1.1. Requirements notation

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

1.2. OSPFv3 LSA Terminology

The TLV-based OSPFv3 LSAs described in this document will be referred to as Extended LSAs. The OSPFv3 fixed-format LSAs [OSPFV3] will be referred to as Legacy LSAs.

2. OSPFv3 Extended LSA Types

In order to provide backward compatibility, new LSA codes must be allocated. There are eight fixed-format LSAs defined in RFC 5340 [OSPFV3]. For ease of implementation and debugging, the LSA function codes are the same as the fixed-format LSAs only with 32, i.e., 0x20, added. The alternative to this mapping was to allocate a bit in the LS Type indicating the new LSA format. However, this would have used one half the LSA function code space for the migration of the eight original fixed-format LSAs. For backward compatibility, the U-bit MUST be set in LS Type so that the LSAs will be flooded by OSPFv3 routers that do not understand them.

LSA function code	LS Type	Description
33	0xA021	E-Router-LSA
34	0xA022	E-Network-LSA
35	0xA023	E-Inter-Area-Prefix-LSA
36	0xA024	E-Inter-Area-Router-LSA
37	0xC025	E-AS-External-LSA
38	N/A	Unused (Not to be allocated)
39	0xA027	E-Type-7-LSA
40	0x8028	E-Link-LSA
41	0xA029	E-Intra-Area-Prefix-LSA

OSPFv3 Extended LSA Types

3. OSPFv3 Extended LSA TLVs

The format of the TLVs within the body of the extended LSAs is the same as the format used by the Traffic Engineering Extensions to OSPF [TE]. The variable TLV section consists of one or more nested Type/Length/Value (TLV) tuples. Nested TLVs are also referred to as sub-TLVs. The format of each TLV is:

```
 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...                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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 0). The TLV is padded to 4-octet alignment; padding is not included in the length field (so a 3-octet value would have a length of 3, but the total size of the TLV would be 8 octets). Nested TLVs are also 32-bit aligned. For example, a 1-byte value would have the length field set to 1, and 3 octets of padding would be added to the end of the value portion of the TLV.

This document defines the following top-level TLV types:

- o 0 - Reserved
- o 1 - Router-Link TLV
- o 2 - Attached-Routers TLV
- o 3 - Inter-Area Prefix TLV
- o 4 - Inter-Area Router TLV
- o 5 - External Prefix TLV
- o 6 - Intra-Area Prefix TLV
- o 7 - IPv6 Link-Local Address TLV
- o 8 - IPv4 Link-Local Address TLV

Additionally, this document defines the following sub-TLV types:

- o 0 - Reserved
- o 1 - IPv6 Forwarding Address sub-TLV
- o 2 - IPv4 Forwarding Address sub-TLV
- o 3 - Route Tag sub-TLV

In general, TLVs and sub-TLVs MAY occur in any order and the specification should define whether the TLV or sub-TLV is required and the behavior when there are multiple occurrences of the TLV or sub-TLV. While this document only describes the usage of TLVs and Sub-TLVs, Sub-TLVs may be nested to any level as long as the Sub-TLVs are fully specified in the specification for the subsuming Sub-TLV.

For backward compatibility, an LSA is not considered malformed from a TLV perspective unless either a required TLV is missing or a specified TLV is less than the minimum required length. Refer to Section 6.3 for more information on TLV backward compatibility.

3.1. Prefix Options Extensions

The prefix options are extended from Appendix A.4.1.1 [OSPFV3]. The applicability of the LA-bit is expanded and it SHOULD be set in Inter-Area-Prefix-TLVs and MAY be set in External-Prefix-TLVs when the advertised host IPv6 address, i.e., PrefixLength = 128, is an interface address. In RFC 5340, the LA-bit is only set in Intra-Area-Prefix-LSAs (Section 4.4.3.9 in [OSPFV3]). This will allow a stable address to be advertised without having to configure a separate loopback address in every OSPFv3 area.

3.1.1. N-bit Prefix Option

Additionally, the N-bit prefix option is defined. The figure below shows the position of the N-bit in the prefix options (pending IANA allocation). This corresponds to the value 0x20.

```
  0  1  2  3  4  5  6  7
+--+--+--+--+--+--+--+--+
|  |  | N|DN| P| x|LA|NU|
+--+--+--+--+--+--+--+--+
```

The Prefix Options field

The N-bit is set in PrefixOptions for a host address (PrefixLength=128) that identifies the advertising router. While it is similar to the LA-bit, there are two differences. The advertising router MAY choose NOT to set the N-bit even when the above conditions are met. If the N-bit is set and the PrefixLength is NOT 128, the N-bit MUST be ignored. Additionally, the N-bit is propagated in the PrefixOptions when an OSPFv3 Area Border Router (ABR) originates an Inter-Area-Prefix-LSA for an Intra-Area route which has the N-bit set in the PrefixOptions. Similarly, the N-bit is propagated in the PrefixOptions when an OSPFv3 NSSA ABR originates an E-AS-External-LSA corresponding to an NSSA route as described in section 3 of RFC 3101 ([NSSA]). The N-bit is added to the Inter-Area-Prefix-TLV (Section 3.4), External-Prefix-TLV (Section 3.6), and Intra-Area-Prefix-TLV (Section 3.7). The N-bit is used as hint to identify the preferred address to reach the advertising OSPFv3 router. This would be in contrast to an Anycast Address [IPV6-ADDRESS-ARCH] which could also be a local address with the LA-bit set. It is useful for applications such as identifying the prefixes corresponding to Node Segment Identifiers (SIDs) in Segment Routing [SEGMENT-ROUTING]. There may be future applications requiring selection of a prefix associated with an OSPFv3 router.

3.2. Router-Link TLV

The Router-Link TLV defines a single router link and the field definitions correspond directly to links in the OSPFv3 Router-LSA, section A.4.3, [OSPFV3]. The Router-Link TLV is only applicable to the E-Router-LSA (Section 4.1). Inclusion in other Extended LSAs MUST be ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          1 (Router-Link)      |       TLV Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |       0       |           Metric              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Interface ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Neighbor Interface ID                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Neighbor Router ID                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                             sub-TLVs                          .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Router-Link TLV

3.3. Attached-Routers TLV

The Attached-Routers TLV defines all the routers attached to an OSPFv3 multi-access network. The field definitions correspond directly to content of the OSPFv3 Network-LSA, section A.4.4, [OSPFV3]. The Attached-Routers TLV is only applicable to the E-Network-LSA (Section 4.2). Inclusion in other Extended LSAs MUST be ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        2 (Attached-Routers)   |       TLV Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Adjacent Neighbor Router ID                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.              Additional Adjacent Neighbors                    .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Attached-Routers TLV

There are two reasons for not having a separate TLV or sub-TLV for each adjacent neighbor. The first is to discourage using the E-Network-LSA for more than its current role of solely advertising the routers attached to a multi-access network. The router’s metric as well as the attributes of individual attached routers should be

advertised in their respective E-Router-LSAs. The second reason is that there is only a single E-Network-LSA per multi-access link with the Link State ID set to the Designated Router’s Interface ID and, consequently, compact encoding has been chosen to decrease the likelihood that the size of the E-Network-LSA will require IPv6 fragmentation when advertised in an OSPFv3 Link State Update packet.

3.4. Inter-Area-Prefix TLV

The Inter-Area-Prefix TLV defines a single OSPFV3 inter-area prefix. The field definitions correspond directly to the content of an OSPFv3 IPv6 Prefix as defined in Section A.4.1, [OSPFV3] and an OSPFv3 Inter-Area-Prefix-LSA, as defined in section A.4.5, [OSPFV3]. Additionally, the PrefixOptions are extended as described in Section 3.1. The Inter-Area-Prefix TLV is only applicable to the E-Inter-Area-Prefix-LSA (Section 4.3). Inclusion in other Extended LSAs MUST be ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       3 (Inter-Area Prefix)   |       TLV Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      0        |                  Metric                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| PrefixLength  | PrefixOptions |              0                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Address Prefix                         |
|                             ...                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            sub-TLVs                           .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Inter-Area Prefix TLV

3.5. Inter-Area-Router TLV

The Inter-Area-Router TLV defines a single OSPFv3 Autonomous System Boundary Router (ASBR) reachable in another area. The field definitions correspond directly to the content of an OSPFv3 Inter-Area-Router-LSA, as defined in section A.4.6, [OSPFV3]. The Inter-Area-Router TLV is only applicable to the E-Inter-Area-Router-LSA (Section 4.4). Inclusion in other Extended LSAs MUST be ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       4 (Inter-Area Router)   |       TLV Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      0        |                Options                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      0        |                Metric                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Destination Router ID                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            sub-TLVs                           .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Inter-Area Router TLV

3.6. External-Prefix TLV

The External-Prefix TLV defines a single OSPFv3 external prefix. With the exception of omitted fields noted below, the field definitions correspond directly to the content of an OSPFv3 IPv6 Prefix as defined in Section A.4.1, [OSPFV3] and an OSPFv3 AS-External-LSA, as defined in section A.4.7, [OSPFV3]. The External-Prefix TLV is only applicable to the E-AS-External-LSA (Section 4.5) and the E-NSSA-LSA (Section 4.6). Additionally, the PrefixOptions are extended as described in Section 3.1. Inclusion in other Extended LSAs MUST be ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       5 (External Prefix)     |       TLV Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         |E| | |                Metric                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| PrefixLength  | PrefixOptions |              0                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Address Prefix                         |
|                             ...                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            sub-TLVs                           .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

External Prefix TLV

In the External-Prefix TLV, the optional IPv6/IPv4 Forwarding Address and External Route Tag are now sub-TLVs. Given the Referenced LS type and Referenced Link State ID from the AS-External-LSA have never been used or even specified, they have been omitted from the External Prefix TLV. If there were ever a requirement for a referenced LSA, it could be satisfied with a sub-TLV.

The following sub-TLVs are defined for optional inclusion in the External Prefix TLV:

- o 1 - IPv6 Forwarding Address sub-TLV (Section 3.10)

- o 2 - IPv4 Forwarding Address sub-TLV (Section 3.11)

- o 3 - Route Tag sub-TLV (Section 3.12)

3.7. Intra-Area-Prefix TLV

The Intra-Area-Prefix TLV defines a single OSPFv3 intra-area prefix. The field definitions correspond directly to the content of an OSPFv3 IPv6 Prefix as defined in Section A.4.1, [OSPFV3] and an OSPFv3 Link-LSA, as defined in section A.4.9, [OSPFV3]. The Intra-Area-Prefix TLV is only applicable to the E-Link-LSA (Section 4.7) and the E-Intra-Area-Prefix-LSA (Section 4.8). Additionally, the PrefixOptions are extended as described in Section 3.1. Inclusion in other Extended LSAs MUST be ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       6 (Intra-Area Prefix)   |       TLV Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      0        |                  Metric                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| PrefixLength  | PrefixOptions |              0                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Address Prefix                         |
|                             ...                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            sub-TLVs                           .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Intra-Area Prefix TLV

3.8. IPv6 Link-Local Address TLV

The IPv6 Link-Local Address TLV is to be used with IPv6 address families as defined in [OSPFV3-AF]. The IPv6 Link-Local Address TLV is only applicable to the E-Link-LSA (Section 4.7). Inclusion in other Extended LSAs MUST be ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  7 (IPv6 Local-Local Address) |       TLV Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-                                                             -+
|                                                               |
+-            IPv6 Link-Local Interface Address                -+
|                                                               |
+-                                                             -+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            sub-TLVs                           .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv6 Link-Local Address TLV

3.9. IPv4 Link-Local Address TLV

The IPv4 Link-Local Address TLV is to be used with IPv4 address families as defined in [OSPFV3-AF]. The IPv4 Link-Local Address TLV is only applicable to the E-Link-LSA (Section 4.7). Inclusion in other Extended LSAs MUST be ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  8 (IPv4 Local-Local Address) |       TLV Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             IPv4 Link-Local Interface Address                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                             sub-TLVs                          .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv4 Link-Local Address TLV

3.10. IPv6-Forwarding-Address Sub-TLV

The IPv6 Forwarding Address TLV has identical semantics to the optional forwarding address in section A.4.7 of [OSPFV3]. The IPv6 Forwarding Address TLV is applicable to the External-Prefix TLV (Section 3.6). Specification as a sub-TLV of other TLVs is not defined herein. The sub-TLV is optional and the first specified instance is used as the Forwarding Address as defined in [OSPFV3]. Instances subsequent to the first MUST be ignored.

The IPv6 Forwarding Address TLV is to be used with IPv6 address families as defined in [OSPFV3-AF] It MUST be ignored for other address families. The IPv6 Forwarding Address TLV length must meet minimum length (16 octets) or it will be considered malformed as described in Section 6.3.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       1 - Forwarding Address  |          sub-TLV Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-                                                             -+
|                                                               |
+-                Forwarding Address                           -+
|                                                               |
+-                                                             -+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv6 Forwarding Address TLV

3.11. IPv4-Forwarding-Address Sub-TLV

The IPv4 Forwarding Address TLV has identical semantics to the optional forwarding address in section A.4.7 of [OSPFV3]. The IPv4 Forwarding Address TLV is The IPv4 Forwarding Address TLV is applicable to the External-Prefix TLV (Section 3.6). Specification as a sub-TLV of other TLVs is not defined herein. The sub-TLV is optional and the first specified instance is used as the Forwarding Address as defined in [OSPFV3]. Instances subsequent to the first MUST be ignored.

The IPv4 Forwarding Address TLV is to be used with IPv4 address families as defined in [OSPFV3-AF] It MUST be ignored for other address families. The IPv4 Forwarding Address TLV length must meet minimum length (4 octets) or it will be considered malformed as described in Section 6.3.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       2 - Forwarding Address  |          sub-TLV Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Forwarding Address                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv4 Forwarding Address TLV

3.12. Route-Tag Sub-TLV

The optional Route Tag sub-TLV has identical semantics to the optional External Route Tag in section A.4.7 of [OSPFV3]. The Route Tag sub-TLV is applicable to the External-Prefix TLV (Section 3.6). Specification as a sub-TLV of other TLVs is not defined herein. The sub-TLV is optional and the first specified instance is used as the Route Tag as defined in [OSPFV3]. Instances subsequent to the first MUST be ignored.

The Route Tag TLV length must meet minimum length (4 octets) or it will be considered malformed as described in Section 6.3.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       3 - Route Tag           |          sub-TLV Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Route Tag                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Route Tag Sub-TLV

4. OSPFv3 Extended LSAs

This section specifies the OSPFv3 Extended LSA formats and encoding. The Extended OSPFv3 LSAs corresponded directly to the original OSPFv3 LSAs specified in [OSPFV3].

4.1. OSPFv3 E-Router-LSA

The E-Router-LSA has an LS Type of 0xA021 and has the same base information content as the Router-LSA defined in section A.4.3 of [OSPFV3]. However, unlike the existing Router-LSA, it is fully extendable and represented as TLVs.

```
 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
+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           LS Age              |1|0|1|         0x21            |
+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Link State ID                           |
+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Advertising Router                        |
+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     LS Sequence Number                        |
+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        LS Checksum            |            Length             |
+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  0  |Nt|x|V|E|B|            Options                           |
+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                             TLVs                              .
.                                                               .
+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Extended Router-LSA

Other than having a different LS Type, all LSA Header fields are the same as defined for the Router-LSA. Initially, only the top-level Router-Link TLV Section 3.2 is applicable and an E-Router-LSA may include multiple Router-Link TLVs. Like the existing Router-LSA, the LSA length is used to determine the end of the LSA including TLVs. Depending on the implementation, it is perfectly valid for an E-Router-LSA to not contain any Router-Link TLVs. However, this would imply that the OSPFv3 router doesn’t have any adjacencies in the corresponding area and is forming an adjacency or adjacencies over unnumbered link(s). Note that no E-Router-LSA stub link is advertised for an unnumbered link.

4.2. OSPFv3 E-Network-LSA

The E-Network-LSA has an LS Type of 0xA022 and has the same base information content as the Network-LSA defined in section A.4.4 of [OSPFV3]. However, unlike the existing Network-LSA, it is fully extendable and represented as TLVs.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          LS Age               |1|0|1|         0x22            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Link State ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Advertising Router                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    LS Sequence Number                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       LS Checksum             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       0       |            Options                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            TLVs                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

E-Network-LSA

Other than having a different LS Type, all LSA Header fields are the same as defined for the Network-LSA. Like the existing Network-LSA, the LSA length is used to determine the end of the LSA including TLVs. Initially, only the top-level Attached-Routers TLV Section 3.3 is applicable. If the Attached-Router TLV is not included in the E-Network-LSA, it is treated as malformed as described in Section 5. Instances of the Attached-Router TLV subsequent to the first MUST be ignored.

4.3. OSPFv3 E-Inter-Area-Prefix-LSA

The E-Inter-Area-Prefix-LSA has an LS Type of 0xA023 and has the same base information content as the Inter-Area-Prefix-LSA defined in section A.4.5 of [OSPFV3]. However, unlike the existing Inter-Area-Prefix-LSA, it is fully extendable and represented as TLVs.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          LS Age               |1|0|1|         0x23            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Link State ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Advertising Router                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   LS Sequence Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       LS Checksum             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                             TLVs                              .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

E-Inter-Area-Prefix-LSA

Other than having a different LS Type, all LSA Header fields are the same as defined for the Inter-Area-Prefix-LSA. In order to retain compatibility and semantics with the current OSPFv3 specification, each Inter-Area-Prefix LSA MUST contain a single Inter-Area Prefix TLV. This will facilitate migration and avoid changes to functions such as incremental SPF computation.

Like the existing Inter-Area-Prefix-LSA, the LSA length is used to determine the end of the LSA including TLV. Initially, only the top-level Inter-Area-Prefix TLV (Section 3.4) is applicable. If the Inter-Area-Prefix TLV is not included in the E-Inter-Area-Prefix-LSA, it is treated as malformed as described in Section 5. Instances of the Inter-Area-Prefix TLV subsequent to the first MUST be ignored.

4.4. OSPFv3 E-Inter-Area-Router-LSA

The E-Inter-Area-Router-LSA has an LS Type of 0xA024 and has the same base information content as the Inter-Area-Router-LSA defined in section A.4.6 of [OSPFV3]. However, unlike the Inter-Area-Router-LSA, it is fully extendable and represented as TLVs.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          LS Age               |1|0|1|         0x24            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Link State ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Advertising Router                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   LS Sequence Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       LS Checksum             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                             TLVs                              .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

E-Inter-Area-Router-LSA

Other than having a different LS Type, all LSA Header fields are the same as defined for the Inter-Area-Router-LSA. In order to retain compatibility and semantics with the current OSPFv3 specification, each Inter-Area-Router LSA MUST contain a single Inter-Area Router TLV. This will facilitate migration and avoid changes to functions such as incremental SPF computation.

Like the existing Inter-Area-Router-LSA, the LSA length is used to determine the end of the LSA including TLV. Initially, only the top-level Inter-Area-Router TLV (Section 3.5) is applicable. If the Inter-Area-Router TLV is not included in the E-Inter-Area-Router-LSA, it is treated as malformed as described in Section 5. Instances of the Inter-Area-Router TLV subsequent to the first MUST be ignored.

4.5. OSPFv3 E-AS-External-LSA

The E-AS-External-LSA has an LS Type of 0xC025 and has the same base information content as the AS-External-LSA defined in section A.4.7 of [OSPFV3]. However, unlike the existing AS-External-LSA, it is fully extendable and represented as TLVs.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          LS Age               |1|1|0|         0x25            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Link State ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Advertising Router                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   LS Sequence Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       LS Checksum             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                             TLVs                              .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

E-AS-External-LSA

Other than having a different LS Type, all LSA Header fields are the same as defined for the AS-External-LSA. In order to retain compatibility and semantics with the current OSPFv3 specification, each LSA MUST contain a single External Prefix TLV. This will facilitate migration and avoid changes to OSPFv3 processes such as incremental SPF computation.

Like the existing AS-External-LSA, the LSA length is used to determine the end of the LSA including sub-TLVs. Initially, only the top-level External-Prefix TLV (Section 3.6) is applicable. If the External-Prefix TLV is not included in the E-External-AS-LSA, it is treated as malformed as described in Section 5. Instances of the External-Prefix TLV subsequent to the first MUST be ignored.

4.6. OSPFv3 E-NSSA-LSA

The E-NSSA-LSA will have the same format and TLVs as the Extended AS-External-LSA Section 4.5. This is the same relationship as exists between the NSSA-LSA defined in section A.4.8 of [OSPFV3], and the AS-External-LSA. The NSSA-LSA will have type 0xA027 which implies area flooding scope. Future requirements may dictate that supported TLVs differ between the E-AS-External-LSA and the E-NSSA-LSA. However, future requirements are beyond the scope of this document.

4.7. OSPFv3 E-Link-LSA

The E-Link-LSA has an LS Type of 0x8028 and will have the same base information content as the Link-LSA defined in section A.4.9 of [OSPFV3]. However, unlike the existing Link-LSA, it is extendable and represented as TLVs.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          LS Age               |1|0|0|         0x28            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Link State ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Advertising Router                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   LS Sequence Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       LS Checksum             |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Rtr Priority  |                Options                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            TLVs                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

E-Link-LSA

Other than having a different LS Type, all LSA Header fields are the same as defined for the Link-LSA.

Only the Intra-Area-Prefix TLV (Section 3.7), IPv6 Link-Local Address TLV (Section 3.8), and IPv4 Link-Local Address TLV (Section 3.9) are applicable to the E-Link-LSA. Like the Link-LSA, the E-Link-LSA affords advertisement of multiple intra-area prefixes. Hence, multiple Intra-Area Prefix TLVs (Section 3.7) may be specified and the LSA length defines the end of the LSA including all TLVs.

A single instance of the IPv6 Link-Local Address TLV (Section 3.8) SHOULD be included in the E-Link-LSA. Instances following the first MUST be ignored. For IPv4 address families as defined in [OSPFV3-AF], this TLV MUST be ignored.

Similarly, only a single instance of the IPv4 Link-Local Address TLV (Section 3.9) SHOULD be included in the E-Link-LSA. Instances

following the first MUST be ignored. For OSPFv3 IPv6 address families as defined in [OSPFV3-AF], this TLV SHOULD be ignored.

If the IPv4/IPv6 Link-Local Address TLV corresponding to the OSPFv3 Address Family is not included in the E-Link-LSA, it is treated as malformed as described in Section 5.

Future specifications may support advertisement of routing and topology information for multiple address families. However, this is beyond the scope of this document.

4.8. OSPFv3 E-Intra-Area-Prefix-LSA

The E-Intra-Area-Prefix-LSA has an LS Type of 0xA029 and has the same base information content as the Intra-Area-Prefix-LSA defined in section A.4.10 of [OSPFV3] except for the Referenced LS Type. However, unlike the Intra-Area-Prefix-LSA, it is fully extendable and represented as TLVs. The Referenced LS Type MUST be either an E-Router-LSA (0xA021) or an E-Network-LSA (0xA022).

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           LS Age              |1|0|1|         0x29            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Link State ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Advertising Router                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   LS Sequence Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        LS Checksum            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       0                       |     Referenced LS Type        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Referenced Link State ID                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Referenced Advertising Router                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            TLVs                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

E-Intra-Area-Prefix-LSA

Other than having a different LS Type, all LSA Header fields are the same as defined for the Intra-Area-Prefix-LSA.

Like the Intra-Area-Prefix-LSA, the E-Intra-Area-Link-LSA affords advertisement of multiple intra-area prefixes. Hence, multiple Intra-Area Prefix TLVs may be specified and the LSA length defines the end of the LSA including all TLVs.

5. Malformed OSPFv3 Extended LSA Handling

Extended LSAs that have inconsistent length or other encoding errors, as described herein, MUST NOT be installed in the Link State Database, acknowledged, or flooded. Reception of malformed LSAs SHOULD be counted and/or logged for examination by the administrator of the OSPFv3 Routing Domain. Note that for the purposes of length validation, a TLV or Sub-TLV should not be considered invalid unless the length exceeds the length of the LSA or does not meet the minimum length requirements. This allows for Sub-TLVs to be added as described in Section 6.3.

Additionally, an LSA MUST be considered malformed if it does not include all of the required TLVs and Sub-TLVs.

6. LSA Extension Backward Compatibility

In the context of this document, backward compatibility is solely related to the capability of an OSPFv3 router to receive, process, and originate the TLV-based LSAs defined herein. Unrecognized TLVs and sub-TLVs are ignored. Backward compatibility for future OSPFv3 extensions utilizing the TLV-based LSAs is out of scope and must be covered in the documents describing those extensions. Both full and, if applicable, partial deployment SHOULD be specified for future TLV-based OSPFv3 LSA extensions.

6.1. Full Extended LSA Migration

If ExtendedLSASupport is enabled Appendix A, OSPFv3 Extended LSAs will be originated and used for the SPF computation. Individual OSPF Areas can be migrated separately with the Legacy AS-External LSAs being originated and used for the SPF computation. This is accomplished by enabled AreaExtendedLSASupport Appendix B.

An OSPFv3 routing domain or area may be non-disruptively migrated using separate OSPFv3 instances for the extended LSAs. Initially, the OSPFv3 instances with ExtendedLSASupport will have a lower preference, i.e., higher administrative distance, than the OSPFv3 instances originating and using the Legacy LSAs. Once the routing domain or area is fully migrated and the OSPFv3 Routing Information Bases (RIB) have been verified, the OSPFv3 instances using the extended LSAs can be given preference. When this has been completed and the routing within the OSPF routing domain or area has been verified, the original OSPFv3 instance using Legacy LSAs can be removed.

6.2. Extended LSA Sparse-Mode Backward Compatibility

In this mode, OSPFv3 will use the Legacy LSAs for the SPF computation and will only originate extended LSAs when LSA origination is required in support of additional functionality. Furthermore, those extended LSAs will only include the top-level TLVs (e.g., Router-Link TLVs or Inter-Area TLVs) which require further specification for that new functionality. However, if a top-level TLV is advertised, it MUST include required Sub-TLVs or it will be considered malformed as described in Section 5. Hence, this mode of compatibility is known as "sparse-mode". The advantage of sparse-mode is that functionality utilizing the OSPFv3 extended LSAs can be added to an existing OSFPv3 routing domain without the requirement for migration. In essence, this compatibility mode is very much like the approach taken for OSPFv2 [OSPF-PREFIX-LINK]. As with all the compatibility modes, backward compatibility for the functions utilizing the extended LSAs must be described in the IETF documents describing those functions.

6.3. LSA TLV Processing Backward Compatibility

This section defines the general rules for processing LSA TLVs. To ensure compatibility of future TLV-based LSA extensions, all implementations MUST adhere to these rules:

1. Unrecognized TLVs and sub-TLVs are ignored when parsing or processing Extended-LSAs.

2. Whether or not partial deployment of a given TLV is supported MUST be specified.

3. If partial deployment is not supported, mechanisms to ensure the corresponding feature are not deployed MUST be specified in the document defining the new TLV or sub-TLV.

4. If partial deployment is supported, backward compatibility and partial deployment MUST be specified in the document defining the new TLV or sub-TLV.

5. If a TLV or Sub-TLV is recognized but the length is less than the minimum, then the LSA should be considered malformed and it SHOULD NOT be acknowledged. Additionally, the occurrence SHOULD be logged with enough information to identify the LSA by type, originator, and sequence number and the TLV or Sub-TLV in error. Ideally, the log entry would include the hexadecimal or binary representation of the LSA including the malformed TLS or Sub-TLV.

6. Documents specifying future TLVs or Sub-TLVs MUST specify the requirements for usage of those TLVs or Sub-TLVs.

7. Future TLV or Sub-TLVs must be optional. However, there may be requirements for Sub-TLVs if an optional TLV is specified.

7. Security Considerations

In general, extendible OSPFv3 LSAs are subject to the same security concerns as those described in RFC 5340 [OSPFV3]. Additionally, implementations must assure that malformed TLV and sub-TLV permutations do not result in errors that cause hard OSPFv3 failures.

If there were ever a requirement to digitally sign OSPFv3 LSAs as described for OSPFv2 LSAs in RFC 2154 [OSPF-DIGITAL-SIGNATURE], the mechanisms described herein would greatly simplify the extension.

8. IANA Considerations

This specification defines nine OSPFv3 Extended LSA types as described in Section 2. These are added the existing OSPFv3 LSA Function Codes registry.

The specification defines a new code point for the N-bit in the OSPFv3 Prefix-Options registry. The value 0x20 is suggested.

This specification also creates two registries OSPFv3 Extended-LSAs TLVs and sub-TLVs. The TLV and sub-TLV code-points in these registries are common to all Extended-LSAs and their respective definitions must define where they are applicable.

8.1. OSPFv3 Extended-LSA TLV Registry

The OSPFv3 Extended-LSA TLV registry defines top-level TLVs for Extended-LSAs and should be placed in the existing OSPFv3 IANA registry.

Nine values are allocated by this specification:

- o 0 - Reserved
- o 1 - Router-Link TLV
- o 2 - Attached-Routers TLV
- o 3 - Inter-Area Prefix TLV
- o 4 - Inter-Area Router TLV
- o 5 - External Prefix TLV

- o 6 - Intra-Area Prefix TLV
- o 7 - IPv6 Link-Local Address TLV
- o 8 - IPv4 Link-Local Address TLV

Types in the range 9-32767 are allocated via IETF Consensus or IESG Approval.

Types in the range 32768-33023 are for experimental use; these will not be registered with IANA, and MUST NOT be mentioned by RFCs.

Types in the range 33024-45055 are to be assigned on a First-Come-First-Serve (FCFS) basis.

Types in the range 45056-65535 are not to be assigned at this time. Before any assignments can be made in the 33024-65535 range, there MUST be an IETF specification that specifies IANA Considerations that covers the range being assigned.

8.2. OSPFv3 Extended-LSA sub-TLV Registry

The OSPFv3 Extended-LSA sub-TLV registry defines sub-TLVs at any level of nesting for Extended-LSAs and should be placed in the existing OSPFv3 IANA registry.

Four values are allocated by this specification:

- o 0 - Reserved
- o 1 - IPv6 Forwarding Address sub-TLV
- o 2 - IPv4 Forwarding Address sub-TLV
- o 3 - Route Tag sub-TLV

Types in the range 4-32767 are allocated via IETF Consensus or IESG Approval.

Types in the range 32768-33023 are for experimental use; these will not be registered with IANA, and MUST NOT be mentioned by RFCs.

Types in the range 33024-45055 are to be assigned on a First-Come-First-Serve (FCFS) basis.

Types in the range 45056-65535 are not to be assigned at this time. Before any assignments can be made in the 33024-65535 range, there

MUST be an IETF specification that specifies IANA Considerations that covers the range being assigned.

9. Contributors

Contributors' Addresses

Sina Mirtorabi
Cisco Systems
170 Tasman Drive
San Jose, CA 95134
USA
Email: sina@cisco.com

10. References

10.1. Normative References

[NSSA] Murphy, P., "The OSPF Not-So-Stubby Area (NSSA) Option", RFC 3101, January 2003.

[OSPFV3] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

[OSPFV3-AF] Lindem, A., Mirtorabi, S., Roy, A., Barnes, M., and R. Aggarwal, "Support of Address Families in OSPFv3", RFC 5838, April 2010.

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.

[TE] Katz, D., Yeung, D., and K. Kompella, "Traffic Engineering Extensions to OSPF", RFC 3630, September 2003.

10.2. Informative References

[IPV6-ADDRESS-ARCH] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[MT-OSPFV3]
Mirtorabi, S. and A. Roy, "Multi-topology routing in OSPFv3 (MT-OSPFV3)", draft-ietf-ospf-mt-ospfv3-04.txt (work in progress), January 2008.

[OSPF-DIGITAL-SIGNATURE]
Murphy, S., Badger, M., and B. Wellington, "OSPF with Digital Signatures", RFC 2154, June 1997.

[OSPF-PREFIX-LINK]
Psenak, P., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPF Prefix/Link Attributes", RFC 7684, December 2015.

[SEGMENT-ROUTING]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-10.txt (work in progress), July 2016.

Appendix A. Appendix A - Global Configuration Parameters

The global configurable parameter ExtendedLSASupport is added to the OSPFv3 protocol. If ExtendedLSASupport is enabled, the OSPFv3 Router will originate OSPFv3 Extended LSAs and use the LSAs for the SPF computation. If ExtendedLSASupport is not enabled, a subset of OSPFv3 Extended LSAs may still be originated and used for other functions as described in Section 6.2.

Appendix B. Appendix B - Area Configuration Parameters

The area configurable parameter AreaExtendedLSASupport is added to the OSPFv3 protocol. If AreaExtendedLSASupport is enabled, the OSPFv3 Router will originate link and area OSPFv3 Extended LSAs and use the LSAs for the SPF computation. Legacy AS-Scoped LSAs will still be originated and used for the AS External LSA computation. If AreaExtendedLSASupport is not enabled a subset of OSPFv3 link and area Extended LSAs may still be originated and used for other functions as described in Section 6.2.

For regular areas, i.e., areas where AS scoped LSAs are flooded, disabling AreaExtendedLSASupport for a regular OSPFv3 area (not a Stub or NSSA area) when ExtendedLSASupport is enabled is contradictory and SHOULD be prohibited by the implementation.

Appendix C. Acknowledgments

OSPFv3 TLV-based LSAs were first proposed in "Multi-topology routing in OSPFv3 (MT-OSPFv3)" [MT-OSPFV3].

Thanks for Peter Psenak for significant contributions to the backward compatibility mechanisms.

Thanks go to Michael Barnes, Mike Dubrovsky, Anton Smirnov, and Tony Przygienda for review of the draft versions and discussions of backward compatibility.

Thanks to Alan Davey for review and comments including the suggestion to separate the extended LSA TLV definitions from the extended LSAs definitions.

Thanks to David Lamparter for review and suggestions on backward compatibility.

Thanks to Karsten Thomann, Chris Bowers, Meng Zhang, and Nagendra Kumar for review and editorial comments.

Thanks to Alia Atlas for substantive Routing Area Director (AD) comments prior to IETF last call.

Thanks to Alvaro Retana and Suresh Krishna for substantive comments during IESG Review.

Thanks to Mehmet Ersue for OPS Directorate review.

The RFC text was produced using Marshall Rose’s xml2rfc tool.

Authors’ Addresses

Acee Lindem
Cisco Systems
301 Midenhall Way
Cary, NC 27513
USA

Email: acee@cisco.com

Abhay Roy
Cisco Systems
170 Tasman Drive
San Jose, CA 95134
USA

Email: akr@cisco.com

Dirk Goethals
Nokia
Copernicuslaan 50
Antwerp 2018
Belgium

Email: dirk.goethals@nokia.com

Veerendranatha Reddy Vallem
Bangalore
India

Email: vallem.veerendra@gmail.com

Fred Baker
Santa Barbara, California 93117
USA

Email: FredBaker.IETF@gmail.com

Open Shortest Path First IGP
Internet-Draft
Intended status: Standards Track
Expires: June 6, 2019

P. Psenak, Ed.
S. Previdi, Ed.
C. Filsfils
Cisco Systems, Inc.
H. Gredler
RtBrick Inc.
R. Shakir
Google, Inc.
W. Henderickx
Nokia
J. Tantsura
Apstra, Inc.
December 3, 2018

OSPF Extensions for Segment Routing
draft-ietf-ospf-segment-routing-extensions-27

Abstract

Segment Routing (SR) allows a flexible definition of end-to-end paths within IGP topologies by encoding paths as sequences of topological sub-paths, called "segments". These segments are advertised by the link-state routing protocols (IS-IS and OSPF).

This draft describes the OSPFv2 extensions required for Segment Routing.

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 will expire on June 6, 2019.

Copyright Notice

Table of Contents

1. Introduction

Segment Routing (SR) allows a flexible definition of end-to-end paths within IGP topologies by encoding paths as sequences of topological sub-paths, called "segments". These segments are advertised by the link-state routing protocols (IS-IS and OSPF). Prefix segments represent an ECMP-aware shortest-path to a prefix (or a node), as per the state of the IGP topology. Adjacency segments represent a hop over a specific adjacency between two nodes in the IGP. A prefix segment is typically a multi-hop path while an adjacency segment, in most cases, is a one-hop path. SR's control-plane can be applied to both IPv6 and MPLS data-planes, and does not require any additional signalling (other than IGP extensions). The IPv6 data plane is out of the scope of this specification - it is not applicable to OSPFv2 which only supports the IPv4 address-family. When used in MPLS networks, SR paths do not require any LDP or RSVP-TE signalling. However, SR can interoperate in the presence of LSPs established with RSVP or LDP.

There are additional segment types, e.g., Binding SID defined in [I-D.ietf-spring-segment-routing].

This draft describes the OSPF extensions required for Segment Routing.

Segment Routing architecture is described in [I-D.ietf-spring-segment-routing].

Segment Routing use cases are described in [RFC7855].

2. Segment Routing Identifiers

Segment Routing defines various types of Segment Identifiers (SIDs): Prefix-SID, Adjacency-SID, LAN Adjacency SID, and Binding SID.

Extended Prefix/Link Opaque LSAs defined in [RFC7684] are used for advertisements of the various SID types.

2.1. SID/Label Sub-TLV

The SID/Label Sub-TLV appears in multiple TLVs or Sub-TLVs defined later in this document. It is used to advertise the SID or label associated with a prefix or adjacency. The SID/Label Sub-TLV has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        SID/Label (variable)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 1

Length: Variable, 3 or 4 octet

SID/Label: If length is set to 3, then the 20 rightmost bits represent a label. If length is set to 4, then the value represents a 32-bit SID.

The receiving router MUST ignore the SID/Label Sub-TLV if the length is other then 3 or 4.

3. Segment Routing Capabilities

Segment Routing requires some additional router capabilities to be advertised to other routers in the area.

These SR capabilities are advertised in the Router Information Opaque LSA (defined in [RFC7770]). The TLVs defined below are applicable to both OSPFv2 and OSPFv3; see also [I-D.ietf-ospf-ospfv3-segment-routing-extensions]

3.1. SR-Algorithm TLV

The SR-Algorithm TLV is a top-level TLV of the Router Information Opaque LSA (defined in [RFC7770]).

The SR-Algorithm TLV is optional. It SHOULD only be advertised once in the Router Information Opaque LSA. If the SR-Algorithm TLV is not advertised by the node, such node is considered as not being segment routing capable.

An SR Router can use various algorithms when calculating reachability to OSPF routers or prefixes in an OSPF area. Examples of these algorithms are metric based Shortest Path First (SPF), various flavors of Constrained SPF, etc. The SR-Algorithm TLV allows a router to advertise the algorithms currently used by the router to other routers in an OSPF area. The SR-Algorithm TLV has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Algorithm 1 | Algorithm...  |   Algorithm n |               |
+-                                                             -+
|                                                               |
+                                                               +
```

where:

Type: 8

Variable, in octets, dependent on number of algorithms advertised.

Algorithm: Single octet identifying the algorithm. The following values are defined by this document:

0: Shortest Path First (SPF) algorithm based on link metric. This is the standard shortest path algorithm as computed by the OSPF protocol. Consistent with the deployed practice for link-state protocols, Algorithm 0 permits any node to overwrite the SPF path with a different path based on its local policy. If the SR-Algorithm TLV is advertised, Algorithm 0 MUST be included.

1: Strict Shortest Path First (SPF) algorithm based on link metric. The algorithm is identical to Algorithm 0 but Algorithm 1 requires that all nodes along the path will honor the SPF routing decision. Local policy at the node claiming support for Algorithm 1 MUST NOT alter the SPF paths computed by Algorithm 1.

When multiple SR-Algorithm TLVs are received from a given router, the receiver MUST use the first occurrence of the TLV in the Router Information LSA. If the SR-Algorithm TLV appears in multiple Router Information LSAs that have different flooding scopes, the SR-Algorithm TLV in the Router Information LSA with the area-scoped flooding scope MUST be used. If the SR-Algorithm TLV appears in

multiple Router Information LSAs that have the same flooding scope, the SR-Algorithm TLV in the Router Information (RI) LSA with the numerically smallest Instance ID MUST be used and subsequent instances of the SR-Algorithm TLV MUST be ignored.

The RI LSA can be advertised at any of the defined opaque flooding scopes (link, area, or Autonomous System (AS)). For the purpose of SR-Algorithm TLV advertisement, area-scoped flooding is REQUIRED.

3.2. SID/Label Range TLV

Prefix SIDs MAY be advertised in a form of an index as described in Section 5. Such index defines the offset in the SID/Label space advertised by the router. The SID/Label Range TLV is used to advertise such SID/Label space.

The SID/Label Range TLV is a top-level TLV of the Router Information Opaque LSA (defined in [RFC7770]).

The SID/Label Range TLV MAY appear multiple times and has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Range Size                |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sub-TLVs (variable)                    |
+-                                                             -+
|                                                               |
+                                                               +
```

where:

Type: 9

Length: Variable, in octets, dependent on Sub-TLVs.

Range Size: 3-octet SID/label range size (i.e., the number of SIDs or labels in the range including the first SID/label). It MUST be greater than 0.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

Initially, the only supported Sub-TLV is the SID/Label Sub-TLV as defined in Section 2.1. The SID/Label Sub-TLV MUST be included in the SID/Label Range TLV. The SID/Label advertised in the SID/Label Sub-TLV represents the first SID/Label in the advertised range.

Only a single SID/Label Sub-TLV MAY be advertised in SID/Label Range TLV. If more then one SID/Label Sub-TLVs are present, the SID/Label Range TLV MUST be ignored.

Multiple occurrences of the SID/Label Range TLV MAY be advertised, in order to advertise multiple ranges. In such case:

- o The originating router MUST encode each range into a different SID/Label Range TLV.

- o The originating router decides the order in which the set of SID/Label Range TLVs are advertised inside the Router Information Opaque LSA. The originating router MUST ensure the order is the same after a graceful restart (using checkpointing, non-volatile storage, or any other mechanism) in order to assure the SID/label range and SID index correspondence is preserved across graceful restarts.

- o The receiving router MUST adhere to the order in which the ranges are advertised when calculating a SID/label from a SID index.

- o The originating router MUST NOT advertise overlapping ranges.

- o When a router receives multiple overlapping ranges, it MUST conform to the procedures defined in [I-D.ietf-spring-segment-routing-mpls].

The following example illustrates the advertisement of multiple ranges:

The originating router advertises the following ranges:

```
Range 1: Range Size: 100   SID/Label Sub-TLV: 100
Range 1: Range Size: 100   SID/Label Sub-TLV: 1000
Range 1: Range Size: 100   SID/Label Sub-TLV: 500
```

The receiving routers concatenate the ranges and build the Segment Routing Global Block (SRGB) as follows:

```
SRGB = [100, 199]
       [1000, 1099]
       [500, 599]
```

The indexes span multiple ranges:

```
index=0 means label 100
...
index 99 means label 199
index 100 means label 1000
index 199 means label 1099
...
index 200 means label 500
...
```

The RI LSA can be advertised at any of the defined flooding scopes (link, area, or autonomous system (AS)). For the purpose of SID/Label Range TLV advertisement, area-scoped flooding is REQUIRED.

3.3. SR Local Block TLV

The SR Local Block TLV (SRLB TLV) contains the range of labels the node has reserved for local SIDs. SIDs from the SRLB MAY be used for Adjacency-SIDs, but also by components other than the OSPF protocol. As an example, an application or a controller can instruct the router to allocate a specific local SID. Some controllers or applications can use the control plane to discover the available set of local SIDs on a particular router. In such cases, the SRLB is advertised in the control plane. The requirement to advertise the SRLB is further described in [I-D.ietf-spring-segment-routing-mpls]. The SRLB TLV is used to advertise the SRLB.

The SRLB TLV is a top-level TLV of the Router Information Opaque LSA (defined in [RFC7770]).

The SRLB TLV MAY appear multiple times in the Router Information Opaque LSA and has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Range Size                |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sub-TLVs (variable)                    |
+-                                                             -+
|                                                               |
+                                                               +
```

where:

Type: 14

Length: Variable, in octets, dependent on Sub-TLVs.

Range Size: 3-octet SID/label range size (i.e., the number of SIDs or labels in the range including the first SID/label). It MUST be greater than 0.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

Initially, the only supported Sub-TLV is the SID/Label Sub-TLV as defined in Section 2.1. The SID/Label Sub-TLV MUST be included in the SRLB TLV. The SID/Label advertised in the SID/Label Sub-TLV represents the first SID/Label in the advertised range.

Only a single SID/Label Sub-TLV MAY be advertised in the SRLB TLV. If more then one SID/Label Sub-TLVs are present, the SRLB TLV MUST be ignored.

The originating router MUST NOT advertise overlapping ranges.

Each time a SID from the SRLB is allocated, it SHOULD also be reported to all components (e.g., controller or applications) in order for these components to have an up-to-date view of the current SRLB allocation. This is required to avoid collisions between allocation instructions.

Within the context of OSPF, the reporting of local SIDs is done through OSPF Sub-TLVs such as the Adjacency-SID (Section 6). However, the reporting of allocated local SIDs can also be done through other means and protocols which are outside the scope of this document.

A router advertising the SRLB TLV MAY also have other label ranges, outside of the SRLB, used for its local allocation purposes which are not advertised in the SRLB TLV. For example, it is possible that an Adjacency-SID is allocated using a local label that is not part of the SRLB.

The RI LSA can be advertised at any of the defined flooding scopes (link, area, or autonomous system (AS)). For the purpose of SRLB TLV advertisement, area-scoped flooding is REQUIRED.

3.4. SRMS Preference TLV

The Segment Routing Mapping Server Preference TLV (SRMS Preference TLV) is used to advertise a preference associated with the node that acts as an SR Mapping Server. The role of an SRMS is described in [I-D.ietf-spring-segment-routing-ldp-interop]. SRMS preference is defined in [I-D.ietf-spring-segment-routing-ldp-interop].

The SRMS Preference TLV is a top-level TLV of the Router Information Opaque LSA (defined in [RFC7770]).

The SRMS Preference TLV MAY only be advertised once in the Router Information Opaque LSA and has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Preference    |                 Reserved                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 15

Length: 4 octets

Preference: 1 octet. SRMS preference value from 0 to 255.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

When multiple SRMS Preference TLVs are received from a given router, the receiver MUST use the first occurrence of the TLV in the Router Information LSA. If the SRMS Preference TLV appears in multiple Router Information LSAs that have different flooding scopes, the SRMS Preference TLV in the Router Information LSA with the narrowest

flooding scope MUST be used. If the SRMS Preference TLV appears in multiple Router Information LSAs that have the same flooding scope, the SRMS Preference TLV in the Router Information LSA with the numerically smallest Instance ID MUST be used and subsequent instances of the SRMS Preference TLV MUST be ignored.

The RI LSA can be advertised at any of the defined flooding scopes (link, area, or autonomous system (AS)). For the purpose of the SRMS Preference TLV advertisement, AS-scoped flooding SHOULD be used. This is because SRMS servers can be located in a different area then consumers of the SRMS advertisements. If the SRMS advertisements from the SRMS server are only used inside the SRMS server's area, area-scoped flooding MAY be used.

4. OSPF Extended Prefix Range TLV

In some cases it is useful to advertise attributes for a range of prefixes. The Segment Routing Mapping Server, which is described in [I-D.ietf-spring-segment-routing-ldp-interop], is an example where we need a single advertisement to advertise SIDs for multiple prefixes from a contiguous address range.

The OSPF Extended Prefix Range TLV, which is a top level TLV of the Extended Prefix LSA described in [RFC7684] is defined for this purpose.

Multiple OSPF Extended Prefix Range TLVs MAY be advertised in each OSPF Extended Prefix Opaque LSA, but all prefix ranges included in a single OSPF Extended Prefix Opaque LSA MUST have the same flooding scope. The OSPF Extended Prefix Range TLV has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix Length |     AF        |         Range Size            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Flags       |                Reserved                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Address Prefix (variable)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Sub-TLVs (variable)                      |
+-                                                             -+
|                                                               |
```

where:

Type: 2

Length: Variable, in octets, dependent on Sub-TLVs.

Prefix length: Length of prefix in bits.

AF: Address family for the prefix. Currently, the only supported value is 0 for IPv4 unicast. The inclusion of address family in this TLV allows for future extension.

Range size: Represents the number of prefixes that are covered by the advertisement. The Range Size MUST NOT exceed the number of prefixes that could be satisfied by the prefix length without including the IPv4 multicast address range (224.0.0.0/3).

Flags: Single octet field. The following flags are defined:

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

where:

IA-Flag: Inter-Area flag. If set, advertisement is of inter-area type. An ABR that is advertising the OSPF Extended Prefix Range TLV between areas MUST set this bit.

This bit is used to prevent redundant flooding of Prefix Range TLVs between areas as follows:

An ABR only propagates an inter-area Prefix Range advertisement from the backbone area to connected non-backbone areas if the advertisement is considered to be the best one. The following rules are used to select the best range from the set of advertisements for the same Prefix Range:

An ABR always prefers intra-area Prefix Range advertisements over inter-area advertisements.

An ABR does not consider inter-area Prefix Range advertisements coming from non-backbone areas.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

Address Prefix: For the address family IPv4 unicast, the prefix itself is encoded as a 32-bit value. The default route is represented by a prefix of length 0. Prefix encoding for other address families is beyond the scope of this specification.

5. Prefix SID Sub-TLV

The Prefix SID Sub-TLV is a Sub-TLV of the OSPF Extended Prefix TLV described in [RFC7684] and the OSPF Extended Prefix Range TLV described in Section 4. It MAY appear more than once in the parent TLV and has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Flags    |   Reserved    |      MT-ID    |    Algorithm  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     SID/Index/Label (variable)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 2

Length: 7 or 8 octets, dependent on the V-flag

Flags: Single octet field. The following flags are defined:

```
  0  1  2  3  4  5  6  7
+--+--+--+--+--+--+--+--+
|  |NP|M |E |V |L |  |  |
+--+--+--+--+--+--+--+--+
```

where:

NP-Flag: No-PHP flag. If set, then the penultimate hop MUST NOT pop the Prefix-SID before delivering packets to the node that advertised the Prefix-SID.

M-Flag: Mapping Server Flag. If set, the SID was advertised by a Segment Routing Mapping Server as described in [I-D.ietf-spring-segment-routing-ldp-interop].

E-Flag: Explicit-Null Flag. If set, any upstream neighbor of the Prefix-SID originator MUST replace the Prefix-SID with the Explicit-NULL label (0 for IPv4) before forwarding the packet.

V-Flag: Value/Index Flag. If set, then the Prefix-SID carries an absolute value. If not set, then the Prefix-SID carries an index.

L-Flag: Local/Global Flag. If set, then the value/index carried by the Prefix-SID has local significance. If not set, then the value/index carried by this Sub-TLV has global significance.

Other bits: Reserved. These MUST be zero when sent and are ignored when received.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

MT-ID: Multi-Topology ID (as defined in [RFC4915]).

Algorithm: Single octet identifying the algorithm the Prefix-SID is associated with as defined in Section 3.1.

A router receiving a Prefix-SID from a remote node and with an algorithm value that such remote node has not advertised in the SR-Algorithm Sub-TLV (Section 3.1) MUST ignore the Prefix-SID Sub-TLV.

SID/Index/Label: According to the V and L flags, it contains:

V-flag is set to 0 and L-flag is set to 0: The SID/Index/Label field is a 4 octet index defining the offset in the SID/Label space advertised by this router

V-flag is set to 1 and L-flag is set to 1: The SID/Index/Label field is a 3 octet local label where the 20 rightmost bits are used for encoding the label value.

All other combinations of V-flag and L-flag are invalid and any SID advertisement received with an invalid setting for V and L flags MUST be ignored.

If an OSPF router advertises multiple Prefix-SIDs for the same prefix, topology and algorithm, all of them MUST be ignored.

When calculating the outgoing label for the prefix, the router MUST take into account, as described below, the E, NP and M flags

advertised by the next-hop router if that router advertised the SID for the prefix. This MUST be done regardless of whether the next-hop router contributes to the best path to the prefix.

The NP-Flag (No-PHP) MUST be set and the E-flag MUST be clear for Prefix-SIDs allocated to inter-area prefixes that are originated by the ABR based on intra-area or inter-area reachability between areas, unless the advertised prefix is directly attached to the ABR.

The NP-Flag (No-PHP) MUST be set and the E-flag MUST be clear for Prefix-SIDs allocated to redistributed prefixes, unless the redistributed prefix is directly attached to the ASBR.

If the NP-Flag is not set, then any upstream neighbor of the Prefix-SID originator MUST pop the Prefix-SID. This is equivalent to the penultimate hop popping mechanism used in the MPLS dataplane. If the NP-flag is not set, then the received E-flag is ignored.

If the NP-flag is set then:

If the E-flag is not set, then any upstream neighbor of the Prefix-SID originator MUST keep the Prefix-SID on top of the stack. This is useful when the originator of the Prefix-SID need to stitch the incoming packet into a continuing MPLS LSP to the final destination. This could occur at an Area Border Router (prefix propagation from one area to another) or at an AS Boundary Router (prefix propagation from one domain to another).

If the E-flag is set, then any upstream neighbor of the Prefix-SID originator MUST replace the Prefix-SID with an Explicit-NULL label. This is useful, e.g., when the originator of the Prefix-SID is the final destination for the related prefix and the originator wishes to receive the packet with the original EXP bits.

When the M-Flag is set, the NP-flag and the E-flag MUST be ignored at reception.

As the Mapping Server does not specify the originator of a prefix advertisement, it is not possible to determine PHP behavior solely based on the Mapping Server advertisement. However, PHP behavior SHOULD be done in following cases:

The Prefix is intra-area type and the downstream neighbor is the originator of the prefix.

The Prefix is inter-area type and downstream neighbor is an ABR, which is advertising prefix reachability and is also generating

the Extended Prefix TLV with the A-flag set for this prefix as described in section 2.1 of [RFC7684].

The Prefix is external type and downstream neighbor is an ASBR, which is advertising prefix reachability and is also generating the Extended Prefix TLV with the A-flag set for this prefix as described in section 2.1 of [RFC7684].

When a Prefix-SID is advertised in an Extended Prefix Range TLV, then the value advertised in the Prefix SID Sub-TLV is interpreted as a starting SID/Label value.

Example 1: If the following router addresses (loopback addresses) need to be mapped into the corresponding Prefix SID indexes:

```
Router-A: 192.0.2.1/32, Prefix-SID: Index 1
Router-B: 192.0.2.2/32, Prefix-SID: Index 2
Router-C: 192.0.2.3/32, Prefix-SID: Index 3
Router-D: 192.0.2.4/32, Prefix-SID: Index 4
```

then the Prefix field in the Extended Prefix Range TLV would be set to 192.0.2.1, Prefix Length would be set to 32, Range Size would be set to 4, and the Index value in the Prefix-SID Sub-TLV would be set to 1.

Example 2: If the following prefixes need to be mapped into the corresponding Prefix-SID indexes:

```
192.0.2.0/30, Prefix-SID: Index 51
192.0.2.4/30, Prefix-SID: Index 52
192.0.2.8/30, Prefix-SID: Index 53
192.0.2.12/30, Prefix-SID: Index 54
192.0.2.16/30, Prefix-SID: Index 55
192.0.2.20/30, Prefix-SID: Index 56
192.0.2.24/30, Prefix-SID: Index 57
```

then the Prefix field in the Extended Prefix Range TLV would be set to 192.0.2.0, Prefix Length would be set to 30, Range Size would be 7, and the Index value in the Prefix-SID Sub-TLV would be set to 51.

6. Adjacency Segment Identifier (Adj-SID)

An Adjacency Segment Identifier (Adj-SID) represents a router adjacency in Segment Routing.

6.1. Adj-SID Sub-TLV

Adj-SID is an optional Sub-TLV of the Extended Link TLV defined in [RFC7684]. It MAY appear multiple times in the Extended Link TLV. The Adj-SID Sub-TLV has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |    Reserved   |   MT-ID       |  Weight       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   SID/Label/Index (variable)                  |
+---------------------------------------------------------------+
```

where:

Type: 2

Length: 7 or 8 octets, dependent on the V flag.

Flags: Single octet field containing the following flags:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|B|V|L|G|P|     |
+-+-+-+-+-+-+-+-+
```

where:

B-Flag: Backup Flag. If set, the Adj-SID refers to an adjacency that is eligible for protection (e.g., using IPFRR or MPLS-FRR) as described in section 3.5 of [I-D.ietf-spring-segment-routing].

The V-Flag: Value/Index Flag. If set, then the Adj-SID carries an absolute value. If not set, then the Adj-SID carries an index.

The L-Flag: Local/Global Flag. If set, then the value/index carried by the Adj-SID has local significance. If not set, then the value/index carried by this Sub-TLV has global significance.

The G-Flag: Group Flag. When set, the G-Flag indicates that the Adj-SID refers to a group of adjacencies (and therefore MAY be assigned to other adjacencies as well).

P-Flag. Persistent flag. When set, the P-Flag indicates that the Adj-SID is persistently allocated, i.e., the Adj-SID value remains consistent across router restart and/or interface flap.

Other bits: Reserved. These MUST be zero when sent and are ignored when received.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

MT-ID: Multi-Topology ID (as defined in [RFC4915].

Weight: Weight used for load-balancing purposes. The use of the weight is defined in [I-D.ietf-spring-segment-routing].

SID/Index/Label: as described in Section 5.

An SR capable router MAY allocate an Adj-SID for each of its adjacencies and set the B-Flag when the adjacency is eligible for protection by an FRR mechanism (IP or MPLS) as described in section 3.5 of [I-D.ietf-spring-segment-routing].

An SR capable router MAY allocate more than one Adj-SID to an adjacency

An SR capable router MAY allocate the same Adj-SID to different adjacencies

When the P-flag is not set, the Adj-SID MAY be persistent. When the P-flag is set, the Adj-SID MUST be persistent.

6.2. LAN Adj-SID Sub-TLV

LAN Adj-SID is an optional Sub-TLV of the Extended Link TLV defined in [RFC7684]. It MAY appear multiple times in the Extended-Link TLV. It is used to advertise a SID/Label for an adjacency to a non-DR router on a broadcast, NBMA, or hybrid [RFC6845] network.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |    Reserved   |     MT-ID     |    Weight     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Neighbor ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    SID/Label/Index (variable)                 |
+---------------------------------------------------------------+
```

where:

Type: 3

Length: 11 or 12 octets, dependent on V-flag.

Flags: same as in Section 6.1

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

MT-ID: Multi-Topology ID (as defined in [RFC4915].

Weight: Weight used for load-balancing purposes. The use of the weight is defined in [I-D.ietf-spring-segment-routing].

Neighbor ID: The Router ID of the neighbor for which the LAN-Adj-SID is advertised.

SID/Index/Label: as described in Section 5.

When the P-flag is not set, the Adj-SID MAY be persistent. When the P-flag is set, the Adj-SID MUST be persistent.

7. Elements of Procedure

7.1. Intra-area Segment routing in OSPFv2

An OSPFv2 router that supports segment routing MAY advertise Prefix-SIDs for any prefix to which it is advertising reachability (e.g., a loopback IP address as described in Section 5).

A Prefix-SID can also be advertised by the SR Mapping Servers (as described in [I-D.ietf-spring-segment-routing-ldp-interop]). A Mapping Server advertises Prefix-SIDs for remote prefixes that exist in the OSPFv2 routing domain. Multiple Mapping Servers can advertise

Prefix-SIDs for the same prefix, in which case the same Prefix-SID MUST be advertised by all of them. The flooding scope of the OSPF Extended Prefix Opaque LSA that is generated by the SR Mapping Server could be either area-scoped or AS-scoped and is determined based on the configuration of the SR Mapping Server.

An SR Mapping Server MUST use the OSPF Extended Prefix Range TLV when advertising SIDs for prefixes. Prefixes of different route-types can be combined in a single OSPF Extended Prefix Range TLV advertised by an SR Mapping Server. Because the OSPF Extended Prefix Range TLV doesn't include a Route-Type field, as in the OSPF Extended Prefix TLV, it is possible to include adjacent prefixes from different Route-Types in the OSPF Extended Prefix Range TLV.

Area-scoped OSPF Extended Prefix Range TLVs are propagated between areas. Similar to propagation of prefixes between areas, an ABR only propagates the OSPF Extended Prefix Range TLV that it considers to be the best from the set it received. The rules used to pick the best OSPF Extended Prefix Range TLV are described in Section 4.

When propagating an OSPF Extended Prefix Range TLV between areas, ABRs MUST set the IA-Flag, that is used to prevent redundant flooding of the OSPF Extended Prefix Range TLV between areas as described in Section 4.

7.2. Inter-area Segment routing in OSPFv2

In order to support SR in a multi-area environment, OSPFv2 MUST propagate Prefix-SID information between areas. The following procedure is used to propagate Prefix SIDs between areas.

When an OSPF ABR advertises a Type-3 Summary LSA from an intra-area prefix to all its connected areas, it will also originate an Extended Prefix Opaque LSA, as described in [RFC7684]. The flooding scope of the Extended Prefix Opaque LSA type will be set to area-local scope. The route-type in the OSPF Extended Prefix TLV is set to inter-area. The Prefix-SID Sub-TLV will be included in this LSA and the Prefix-SID value will be set as follows:

The ABR will look at its best path to the prefix in the source area and find the advertising router associated with the best path to that prefix.

The ABR will then determine if such router advertised a Prefix-SID for the prefix and use it when advertising the Prefix-SID to other connected areas.

If no Prefix-SID was advertised for the prefix in the source area by the router that contributes to the best path to the prefix, the originating ABR will use the Prefix-SID advertised by any other router when propagating the Prefix-SID for the prefix to other areas.

When an OSPF ABR advertises Type-3 Summary LSAs from an inter-area route to all its connected areas, it will also originate an Extended Prefix Opaque LSA, as described in [RFC7684]. The flooding scope of the Extended Prefix Opaque LSA type will be set to area-local scope. The route-type in OSPF Extended Prefix TLV is set to inter-area. The Prefix-SID Sub-TLV will be included in this LSA and the Prefix-SID will be set as follows:

The ABR will look at its best path to the prefix in the backbone area and find the advertising router associated with the best path to that prefix.

The ABR will then determine if such router advertised a Prefix-SID for the prefix and use it when advertising the Prefix-SID to other connected areas.

If no Prefix-SID was advertised for the prefix in the backbone area by the ABR that contributes to the best path to the prefix, the originating ABR will use the Prefix-SID advertised by any other router when propagating the Prefix-SID for the prefix to other areas.

7.3. Segment Routing for External Prefixes

Type-5 LSAs are flooded domain wide. When an ASBR, which supports SR, generates Type-5 LSAs, it SHOULD also originate Extended Prefix Opaque LSAs, as described in [RFC7684]. The flooding scope of the Extended Prefix Opaque LSA type is set to AS-wide scope. The route-type in the OSPF Extended Prefix TLV is set to external. The Prefix-SID Sub-TLV is included in this LSA and the Prefix-SID value will be set to the SID that has been reserved for that prefix.

When an NSSA [RFC3101] ABR translates Type-7 LSAs into Type-5 LSAs, it SHOULD also advertise the Prefix-SID for the prefix. The NSSA ABR determines its best path to the prefix advertised in the translated Type-7 LSA and finds the advertising router associated with that path. If the advertising router has advertised a Prefix-SID for the prefix, then the NSSA ABR uses it when advertising the Prefix-SID for the Type-5 prefix. Otherwise, the Prefix-SID advertised by any other router will be used.

7.4. Advertisement of Adj-SID

The Adjacency Segment Routing Identifier (Adj-SID) is advertised using the Adj-SID Sub-TLV as described in Section 6.

7.4.1. Advertisement of Adj-SID on Point-to-Point Links

An Adj-SID MAY be advertised for any adjacency on a P2P link that is in neighbor state 2-Way or higher. If the adjacency on a P2P link transitions from the FULL state, then the Adj-SID for that adjacency MAY be removed from the area. If the adjacency transitions to a state lower then 2-Way, then the Adj-SID advertisement MUST be withdrawn from the area.

7.4.2. Adjacency SID on Broadcast or NBMA Interfaces

Broadcast, NBMA, or hybrid [RFC6845] networks in OSPF are represented by a star topology where the Designated Router (DR) is the central point to which all other routers on the broadcast, NBMA, or hybrid network connect. As a result, routers on the broadcast, NBMA, or hybrid network advertise only their adjacency to the DR. Routers that do not act as DR do not form or advertise adjacencies with each other. They do, however, maintain 2-Way adjacency state with each other and are directly reachable.

When Segment Routing is used, each router on the broadcast, NBMA, or hybrid network MAY advertise the Adj-SID for its adjacency to the DR using the Adj-SID Sub-TLV as described in Section 6.1.

SR capable routers MAY also advertise a LAN-Adj-SID for other neighbors (e.g., BDR, DR-OTHER) on the broadcast, NBMA, or hybrid network using the LAN-ADJ-SID Sub-TLV as described in Section 6.2.

8. IANA Considerations

This specification updates several existing OSPF registries.

8.1. OSPF Router Information (RI) TLVs Registry

- o 8 (IANA Preallocated) - SR-Algorithm TLV
- o 9 (IANA Preallocated) - SID/Label Range TLV
- o 14 - SR Local Block TLV
- o 15 - SRMS Preference TLV

8.2. OSPFv2 Extended Prefix Opaque LSA TLVs Registry

Following values are allocated:

- o 2 - OSPF Extended Prefix Range TLV

8.3. OSPFv2 Extended Prefix TLV Sub-TLVs Registry

Following values are allocated:

- o 1 - SID/Label Sub-TLV
- o 2 - Prefix SID Sub-TLV

8.4. OSPFv2 Extended Link TLV Sub-TLVs Registry

Following initial values are allocated:

- o 1 - SID/Label Sub-TLV
- o 2 - Adj-SID Sub-TLV
- o 3 - LAN Adj-SID/Label Sub-TLV

8.5. IGP Algorithm Type Registry

IANA is requested to set up a registry called "IGP Algorithm Type" under a new category of "Interior Gateway Protocol (IGP) Parameters" IANA registries. The registration policy for this registry is "Standards Action" ([RFC8126] and [RFC7120]).

Values in this registry come from the range 0-255.

The initial values in the IGP Algorithm Type registry are:

0: Shortest Path First (SPF) algorithm based on link metric. This is the standard shortest path algorithm as computed by the IGP protocol. Consistent with the deployed practice for link-state protocols, Algorithm 0 permits any node to overwrite the SPF path with a different path based on its local policy.

1: Strict Shortest Path First (SPF) algorithm based on link metric. The algorithm is identical to Algorithm 0 but Algorithm 1 requires that all nodes along the path will honor the SPF routing decision. Local policy at the node claiming support for Algorithm 1 MUST NOT alter the SPF paths computed by Algorithm 1.

9. Implementation Status

An implementation survey with seven questions related to the implementer's support of OSPFv2 Segment Routing was sent to the OSPF WG list and several known implementers. This section contains responses from three implementers who completed the survey. No external means were used to verify the accuracy of the information submitted by the respondents. The respondents are considered experts on the products they reported on. Additionally, responses were omitted from implementers who indicated that they have not implemented the function yet.

This section will be removed before publication as an RFC.

Responses from Nokia (former Alcatel-Lucent):

Link to a web page describing the implementation:
https://infoproducts.alcatel-lucent.com/cgi-bin/dbaccessfilename.cgi/3HE10799AAAATQZZA01_V1_7450%20ESS%207750%20SR%20and%207950%20XRS%20Unicast%20Routing%20Protocols%20Guide%20R14.0.R1.pdf

The implementation's level of maturity: Production.

Coverage: We have implemented all sections and have support for the latest draft.

Licensing: Part of the software package that needs to be purchased.

Implementation experience: Great spec. We also performed inter-operability testing with Cisco's OSPF Segment Routing implementation.

Contact information: wim.henderickx@nokia.com

Responses from Cisco Systems:

Link to a web page describing the implementation:

http://www.segment-routing.net/home/tutorial

The implementation's level of maturity: Production.

Coverage: All sections have been implemented according to the latest draft.

Licensing: Part of a commercial software package.

Implementation experience: Many aspects of the draft are result of the actual implementation experience, as the draft evolved from its

initial version to the current one. Interoperability testing with Alcatel-Lucent was performed, which confirmed the draft's ability to serve as a reference for the implementors.

Contact information: ppsenak@cisco.com

Responses from Juniper:

The implementation's name and/or a link to a web page describing the implementation:

Feature name is OSPF SPRING

The implementation's level of maturity: To be released in 16.2 (second half of 2016)

Coverage: All sections implemented except Sections 4, and 6.

Licensing: JUNOS Licensing needed.

Implementation experience: NA

Contact information: shraddha@juniper.net

10. Security Considerations

With the OSPFv2 segment routing extensions defined herein, OSPFv2 will now program the MPLS data plane [RFC3031] in addition to the IP data plane. Previously, LDP [RFC5036] or another label distribution mechanism was required to advertise MPLS labels and program the MPLS data plane.

In general, the same types of attacks that can be carried out on the IP control plane can be carried out on the MPLS control plane resulting in traffic being misrouted in the respective data planes. However, the latter can be more difficult to detect and isolate.

Existing security extensions as described in [RFC2328] and [RFC7684] apply to these segment routing extensions. While OSPF is under a single administrative domain, there can be deployments where potential attackers have access to one or more networks in the OSPF routing domain. In these deployments, stronger authentication mechanisms such as those specified in [RFC7474] SHOULD be used.

Implementations MUST assure that malformed TLV and Sub-TLV defined in this document are detected and do not provide a vulnerability for attackers to crash the OSPFv2 router or routing process. Reception of malformed TLV or Sub-TLV SHOULD be counted and/or logged for

further analysis. Logging of malformed TLVs and Sub-TLVs SHOULD be rate-limited to prevent a Denial of Service (DoS) attack (distributed or otherwise) from overloading the OSPF control plane.

11. Contributors

The following people gave a substantial contribution to the content of this document: Acee Lindem, Ahmed Bashandy, Martin Horneffer, Bruno Decraene, Stephane Litkowski, Igor Milojevic, Rob Shakir and Saku Ytti.

12. Acknowledgements

We would like to thank Anton Smirnov for his contribution.

Thanks to Acee Lindem for the detail review of the draft, corrections, as well as discussion about details of the encoding.

13. References

13.1. Normative References

[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.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-15 (work in progress), October 2018.

[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>.

[RFC3101] Murphy, P., "The OSPF Not-So-Stubby Area (NSSA) Option", RFC 3101, DOI 10.17487/RFC3101, January 2003, <https://www.rfc-editor.org/info/rfc3101>.

[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>.

[RFC6845] Sheth, N., Wang, L., and J. Zhang, "OSPF Hybrid Broadcast and Point-to-Multipoint Interface Type", RFC 6845, DOI 10.17487/RFC6845, January 2013, <https://www.rfc-editor.org/info/rfc6845>.

[RFC7120] Cotton, M., "Early IANA Allocation of Standards Track Code Points", BCP 100, RFC 7120, DOI 10.17487/RFC7120, January 2014, <https://www.rfc-editor.org/info/rfc7120>.

[RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute Advertisement", RFC 7684, DOI 10.17487/RFC7684, November 2015, <https://www.rfc-editor.org/info/rfc7684>.

[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>.

[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>.

13.2. Informative References

[I-D.ietf-ospf-ospfv3-segment-routing-extensions] Psenak, P. and S. Previdi, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-18 (work in progress), November 2018.

[RFC7474] Bhatia, M., Hartman, S., Zhang, D., and A. Lindem, Ed., "Security Extension for OSPFv2 When Using Manual Key Management", RFC 7474, DOI 10.17487/RFC7474, April 2015, <https://www.rfc-editor.org/info/rfc7474>.

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <https://www.rfc-editor.org/info/rfc7855>.

Authors' Addresses

Peter Psenak (editor)
Cisco Systems, Inc.
Apollo Business Center
Mlynske nivy 43
Bratislava 821 09
Slovakia

Email: ppsenak@cisco.com

Stefano Previdi (editor)
Cisco Systems, Inc.
Via Del Serafico, 200
Rome 00142
Italy

Email: stefano@previdi.net

Clarence Filsfils
Cisco Systems, Inc.
Brussels
Belgium

Email: cfilsfil@cisco.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com

Rob Shakir
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
US

Email: robjs@google.com

Wim Henderickx
Nokia
Copernicuslaan 50
Antwerp 2018
BE

Email: wim.henderickx@nokia.com

Jeff Tantsura
Apstra, Inc.

Email: jefftant.ietf@gmail.com

Open Shortest Path First IGP
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2015

P. Psenak, Ed.
S. Previdi, Ed.
C. Filsfils
Cisco Systems, Inc.
H. Gredler
Juniper Networks, Inc.
R. Shakir
British Telecom
W. Henderickx
Alcatel-Lucent
J. Tantsura
Ericsson
July 2, 2014

OSPFv3 Extensions for Segment Routing
draft-psenak-ospf-segment-routing-ospfv3-extension-02

Abstract

Segment Routing (SR) allows for a flexible definition of end-to-end paths within IGP topologies by encoding paths as sequences of topological sub-paths, called "segments". These segments are advertised by the link-state routing protocols (IS-IS and OSPF).

This draft describes the necessary OSPFv3 extensions that need to be introduced for Segment Routing.

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 will expire on January 3, 2015.

Copyright Notice

Table of Contents

1. Introduction

Segment Routing (SR) allows for a flexible definition of end-to-end paths within IGP topologies by encoding paths as sequences of topological sub-paths, called "segments". These segments are advertised by the link-state routing protocols (IS-IS and OSPF). Prefix segments represent an ecmp-aware shortest-path to a prefix (or a node), as per the state of the IGP topology. Adjacency segments represent a hop over a specific adjacency between two nodes in the IGP. A prefix segment is typically a multi-hop path while an adjacency segment, in most of the cases, is a one-hop path. SR's control-plane can be applied to both IPv6 and MPLS data-planes, and do not require any additional signaling (other than the regular IGP). For example, when used in MPLS networks, SR paths do not require any LDP or RSVP-TE signaling. Still, SR can interoperate in the presence of LSPs established with RSVP or LDP .

This draft describes the necessary OSPFv3 extensions that need to be introduced for Segment Routing.

Segment Routing architecture is described in [I-D.filsfils-rtgwg-segment-routing].

Segment Routing use cases are described in [I-D.filsfils-rtgwg-segment-routing-use-cases].

2. Segment Routing Identifiers

Segment Routing defines various types of Segment Identifiers (SIDs): Prefix-SID, Adjacency-SID, LAN Adjacency SID and Binding SID.

2.1. SID/Label sub-TLV

SID/Label sub-TLV appears in multiple TLVs or Sub-TLVs defined later in this document. It is used to advertise SID or label associated with the prefix or adjacency. SID/Label TLV has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Type            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      SID/Label (variable)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: TBD, suggested value 1

Length: variable, 3 or 4 bytes

SID/Label: if length is set to 3, then the 20 rightmost bits represent a label. If length is set to 4 then the value represents a 32 bit SID.

The receiving router MUST ignore SID/Label sub-TLV if the length is other then 3 or 4.

3. Segment Routing Capabilities

Segment Routing requires some additional capabilities of the router to be advertised to other routers in the area.

These SR capabilities are advertised in OSPFv3 Router Information Opaque LSA (defined in [RFC4970]).

3.1. SR-Algorithm TLV

SR-Algorithm TLV is a TLV of Router Information Opaque LSA (defined in [RFC4970]).

Router may use various algorithms when calculating reachability to other nodes in area or to prefixes attached to these nodes. Examples of these algorithms are metric based Shortest Path First (SPF), various sorts of Constrained SPF, etc. SR-Algorithm TLV allows a router to advertise algorithms that router is currently using to other routers in an area. SR-Algorithm TLV has following structure:

```
 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            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Algorithm 1  |  Algorithm... |  Algorithm n |                |
+-                                                             -+
|                                                               |
+                                                               +
```

where:

Type: TBD, suggested value 8

Length: variable

Algorithm: one octet identifying the algorithm. The following value has been defined:

0: IGP metric based SPT.

RI LSA can be advertised at any of the defined flooding scopes (link, area, or autonomous system (AS)). For the purpose of the SR-Algorithm TLV propagation area scope flooding is required.

3.2. SID/Label Range TLV

The SID/Label Range TLV is a TLV of Router Information Opaque LSA (defined in [RFC4970]).

SID/Label Sub-TLV MAY appear multiple times and has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Type          |               Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Range Size               |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Sub-TLVs (variable)                       |
+-                                                             -+
|                                                               |
+                                                               +
```

where:

Type: TBD, suggested value 9

Length: variable

Range Size: 3 octets of SID/label range

Currently the only supported Sub-TLV is the SID/Label TLV as defined in Section 2.1. SID/Label advertised in SID/Label TLV represents the first SID/Label from the advertised range.

Multiple occurrence of the SID/Label Range TLV MAY be advertised, in order to advertise multiple ranges. In such case:

- o The originating router MUST encode each range into a different SID/Label Range TLV.

- o The originating router decides in which order the set of SID/Label Range TLVs are advertised inside Router Information Opaque LSA. The originating router MUST ensure the order is same after a graceful restart (using checkpointing, non-volatile storage or any other mechanism) in order to guarantee the same order before and after graceful restart.

- o Receiving router must adhere to the order in which the ranges are advertised when calculating a SID/label from the SID index.

- o A router not supporting multiple occurrences SID/Label Range TLV MUST take into consideration the first occurrence in the received set.

Here follows an example of advertisement of multiple ranges:

```
The originating router advertises following ranges:
   Range 1: [100, 199]
   Range 2: [1000, 1099]
   Range 3: [500, 599]

The receiving routers concatenate the ranges and build the SRGB
is as follows:

SRGB = [100, 199]
       [1000, 1099]
       [500, 599]

The indexes span multiple ranges:

   index=0 means label 100
   ...
   index 99 means label 199
   index 100 means label 1000
   index 199 means label 1099
   ...
   index 200 means label 500
   ...
```

RI LSA can be advertised at any of the defined flooding scopes (link, area, or autonomous system (AS)). For the purpose of the SR-Capability TLV propagation area scope flooding is required.

4. Prefix SID Identifier

A new extended OSPFv3 LSAs as defined in [I-D.ietf-ospf-ospfv3-lsa-extend] are used to advertise SID or label values associated with the prefix in OSPFv3.

4.1. Prefix SID Sub-TLV

The Prefix SID Sub-TLV is a Sub-TLV of the following OSPFv3 TLVs as defined in [I-D.ietf-ospf-ospfv3-lsa-extend]:

Intra-Area Prefix TLV

Inter-Area Prefix TLV

External Prefix TLV

It MAY appear more than once and has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Type            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags       |  Algorithm  |          Range Size           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       SID/Index/Label (variable)              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: TBD, suggested value 2.

Length: variable

Flags: 1 octet field. The following flags are defined:

```
 0
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|N|P|M|E|V|L|   |
+-+-+-+-+-+-+-+-+
```

where:

N-Flag: Node-SID flag. If set, then the Prefix-SID refers to the router identified by the prefix. Typically, the N-Flag is set on Prefix-SIDs attached to a router loopback address. The N-Flag is set when the Prefix-SID is a Node- SID as described in [I-D.filsfils-rtgwg-segment-routing].

P-Flag: no-PHP flag. If set, then the penultimate hop MUST NOT pop the Prefix-SID before delivering the packet to the node that advertised the Prefix-SID.

M-Flag: Mapping Server Flag. If set, the SID is advertised from the Segment Routing Mapping Server functionality as described in [I-D.filsfils-rtgwg-segment-routing-use-cases].

E-Flag: Explicit-Null Flag. If set, any upstream neighbor of the Prefix-SID originator MUST replace the Prefix-SID with a Prefix-SID having an Explicit-NULL value (0 for IPv4) before forwarding the packet.

The V-Flag: Value/Index Flag. If set, then the Prefix-SID carries an absolute value. If not set, then the Prefix-SID carries an index.

The L-Flag: Local/Global Flag. If set, then the value/index carried by the PrefixSID has local significance. If not set, then the value/index carried by this subTLV has global significance.

Other bits: MUST be zero when sent and ignored when received.

Algorithm: one octet identifying the algorithm the Prefix-SID is associated with as defined in Section 3.1.

Range Size: this field provides the ability to specify a range of addresses and their associated Prefix SIDs. It represents a compression scheme to distribute a continuous Prefix and their continuous, corresponding SID/Label Block. If a single SID is advertised then the Range Size field MUST be set to 1. For range advertisements > 1, Range Size represents the number of addresses that need to be mapped into a Prefix-SID.

SID/Index/Label: label or index value depending on the V-bit setting.

Examples:

A 32 bit global index defining the offset in the SID/Label space advertised by this router - in this case the V and L flags MUST be unset.

A 24 bit local label where the 20 rightmost bits are used for encoding the label value - in this case the V and L flags MUST be set.

If multiple Prefix-SIDs are advertised for the same prefix, the receiving router MUST use the first encoded SID and MAY use the subsequent ones.

When propagating Prefix-SIDs between areas, if multiple prefix-SIDs are advertised for a prefix, an implementation SHOULD preserve the original ordering, when advertising prefix-SIDs to other areas. This allows implementations that only use single Prefix-SID to have a consistent view across areas.

When calculating the outgoing label for the prefix, the router MUST take into account E and P flags advertised by the next-hop router, if next-hop router advertised the SID for the prefix. This MUST be done regardless of next-hop router contributing to the best path to the prefix or not.

P-Flag (no-PHP) MUST be set on the Prefix-SIDs allocated to inter-area prefixes that are originated by the ABR based on intra-area or inter-area reachability between areas. In case the inter-area prefix is generated based on the prefix which is directly attached to the ABR, P-Flag SHOULD NOT be set

P-Flag (no-PHP) MUST NOT be set on the Prefix-SIDs allocated to redistributed prefixes, unless the redistributed prefix is directly attached to ASBR, in which case the P-Flag SHOULD NOT be set.

If the P-flag is not set then any upstream neighbor of the Prefix-SID originator MUST pop the Prefix-SID. This is equivalent to the penultimate hop popping mechanism used in the MPLS dataplane. In such case MPLS EXP bits of the Prefix-SID are not preserved to the ultimate hop (the Prefix-SID being removed). If the P-flag is unset the received E-flag is ignored.

If the P-flag is set then:

If the E-flag is not set then any upstream neighbor of the Prefix-SID originator MUST keep the Prefix-SID on top of the stack. This is useful when the originator of the Prefix-SID must stitch the incoming packet into a continuing MPLS LSP to the final destination. This could occur at an inter-area border router (prefix propagation from one area to another) or at an inter-domain border router (prefix propagation from one domain to another).

If the E-flag is set then any upstream neighbor of the Prefix-SID originator MUST replace the PrefixSID with a Prefix-SID having an Explicit-NULL value. This is useful, e.g., when the originator of the Prefix-SID is the final destination for the related prefix and the originator wishes to receive the packet with the original EXP bits.

When M-Flag is set, P-flag MUST be set and E-bit MUST NOT be set.

Example 1: if the following router addresses (loopback addresses) need to be mapped into the corresponding Prefix SID indexes:

```
Router-A: 192::1/128, Prefix-SID: Index 1
Router-B: 192::2/128, Prefix-SID: Index 2
Router-C: 192::3/128, Prefix-SID: Index 3
Router-D: 192::4/128, Prefix-SID: Index 4
```

then the Address Prefix field in Intra-Area Prefix TLV, Inter-Area Prefix TLV or External Prefix TLV is set to 192::1, Prefix Length in

these TLVs would be set to 128, Range Size in Prefix SID sub-TLV would be set to 4 and Index value would be set to 1.

Example 2: If the following prefixes need to be mapped into the corresponding Prefix-SID indexes:

```
10:1:1::0/120,   Prefix-SID: Index 51
10:1:1::100/120, Prefix-SID: Index 52
10:1:1::200/120, Prefix-SID: Index 53
10:1:1::300/120, Prefix-SID: Index 54
10:1:1::400/120, Prefix-SID: Index 55
10:1:1::500/120, Prefix-SID: Index 56
10:1:1::600/120, Prefix-SID: Index 57
```

then the Address Prefix field in Intra-Area Prefix TLV, Inter-Area Prefix TLV or External Prefix TLV is set to 10:1:1::0, Prefix Length in these TLVs would be set to 120, Range Size in Prefix SID sub-TLV would be set to 7 and Index value would be set to 51.

4.2. SID/Label Binding sub-TLV

SID/Label Binding sub-TLV is used to advertise SID/Label mapping for a path to the prefix.

The SID/Label Binding TLV MAY be originated by any router in an OSPFv3 domain. The router may advertise a SID/Label binding to a FEC along with at least a single 'nexthop style' anchor. The protocol supports more than one 'nexthop style' anchor to be attached to a SID/Label binding, which results into a simple path description language. In analogy to RSVP the terminology for this is called an 'Explicit Route Object' (ERO). Since ERO style path notation allows to anchor SID/label bindings to both link and node IP addresses any label switched path, can be described. Furthermore also SID/Label Bindings from external protocols can get easily re-advertised.

The SID/Label Binding TLV may be used for advertising SID/Label Bindings and their associated Primary and Backup paths. In one single TLV either a primary ERO Path, a backup ERO Path or both are advertised. If a router wants to advertise multiple parallel paths then it can generate several TLVs for the same Prefix/FEC. Each occurrence of a Binding TLV with respect with a given FEC Prefix has accumulating and not canceling semantics.

SID/Label Binding sub-TLV is a sub-TLV of the following OSPFv3 TLVs, as defined in [I-D.ietf-ospf-ospfv3-lsa-extend]:

Intra-Area Prefix TLV

Inter-Area Prefix TLV

External Prefix TLV

Multiple SID/Label Binding sub-TLVs can be present in above mentioned TLVs. SID/Label Binding sub-TLV has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags       |   Weight    |           Range Size          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Sub-TLVs (variable)                     |
+-                                                             -+
|                                                               |
```

where:

Type: TBD, suggested value 5

Length: variable

Flags: 1 octet field of following flags:

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

where:

M-bit - When the bit is set the binding represents the mirroring context as defined in [I-D.minto-rsvp-lsp-egress-fast-protection].

Weight: weight used for load-balancing purposes. The use of the weight is defined in [I-D.filsfils-rtgwg-segment-routing].

Range Size: usage is the same as described in Section 4.1

SID/Label Binding sub-TLV currently supports following Sub-TLVs:

SID/Label sub-TLV as described in Section 2.1. This sub-TLV MUST appear in the SID/Label Binding Sub-TLV and it MUST only appear once.

ERO Metric sub-TLV as defined in Section 4.2.1.

ERO sub-TLVs as defined in Section 4.2.2.

4.2.1. ERO Metric sub-TLV

ERO Metric sub-TLV is a Sub-TLV of the SID/Label Binding TLV.

The ERO Metric sub-TLV carries the cost of an ERO path. It is used to compare the cost of a given source/destination path. A router SHOULD advertise the ERO Metric sub-TLV. The cost of the ERO Metric sub-TLV SHOULD be set to the cumulative IGP or TE path cost of the advertised ERO. Since manipulation of the Metric field may attract or distract traffic from and to the advertised segment it MAY be manually overridden.

```
 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            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Metric (4 octets)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

ERO Metric sub-TLV format

where:

Type: TBD, suggested value 6

Length: 4 bytes

Metric: 4 bytes

4.2.2. ERO sub-TLVs

All ’ERO’ information represents an ordered set which describes the segments of a path. The last ERO sub-TLV describes the segment closest to the egress point, contrary the first ERO sub-TLV describes the first segment of a path. If a router extends or stitches a path it MUST prepend the new segments path information to the ERO list.

The above similarly applies to backup EROs.

All ERO Sub-TLVs must immediately follow the (SID)/Label Sub-TLV.

All Backup ERO sub-TLVs must immediately follow last ERO Sub-TLV.

4.2.2.1. IPv4 ERO sub-TLV

IPv4 ERO sub-TLV is a sub-TLV of the SID/Label Binding sub-TLV.

The IPv4 ERO sub-TLV describes a path segment using IPv4 Address style of encoding. Its semantics have been borrowed from [RFC3209].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Type            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Flags      |                   Reserved                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       IPv4 Address (4 octets)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv4 ERO sub-TLV format

where:

Type: TBD, suggested value 7

Length: 8 bytes

Flags: 1 octet field of following flags:

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

where:

L-bit - If the L bit is set, then the value of the attribute is 'loose.' Otherwise, the value of the attribute is 'strict.'

IPv4 Address - the address of the explicit route hop.

4.2.2.2. IPv6 ERO sub-TLV

IPv6 ERO sub-TLV is a sub-TLV of the SID/Label Binding sub-TLV.

The IPv6 ERO sub-TLV (Type TBA) describes a path segment using IPv6 Address style of encoding. Its semantics have been borrowed from [RFC3209].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Flags      |                   Reserved                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-                                                             -+
|                                                               |
+-                          IPv6 Address                       -+
|                                                               |
+-                                                             -+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv6 ERO sub-TLV format

where:

Type: TBD, suggested value 8

Length: 8 bytes

Flags: 1 octet field of following flags:

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

where:

L-bit - If the L bit is set, then the value of the attribute is ’loose.’ Otherwise, the value of the attribute is ’strict.’

IPv6 Address - the address of the explicit route hop.

4.2.2.3. Unnumbered Interface ID ERO sub-TLV

Unnumbered Interface ID ERO sub-TLV is a sub-TLV of the SID/Label Binding sub-TLV.

The appearance and semantics of the ’Unnumbered Interface ID’ have been borrowed from [RFC3477].

The Unnumbered Interface-ID ERO sub-TLV describes a path segment that spans over an unnumbered interface. Unnumbered interfaces are

referenced using the interface index. Interface indices are assigned local to the router and therefore not unique within a domain. All elements in an ERO path need to be unique within a domain and hence need to be disambiguated using a domain unique Router-ID.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Type            |              Length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |                  Reserved                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Router ID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Interface ID                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Unnumbered Interface ID ERO sub-TLV format

Type: TBD, suggested value 9

Length: 12 bytes

Flags: 1 octet field of following flags:

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

where:

L-bit - If the L bit is set, then the value of the attribute is 'loose.' Otherwise, the value of the attribute is 'strict.'

Router-ID: Router-ID of the next-hop.

Interface ID: is the identifier assigned to the link by the router specified by the Router-ID.

4.2.2.4. IPv4 Backup ERO sub-TLV

IPv4 Prefix Backup ERO sub-TLV is a sub-TLV of the SID/Label Binding sub-TLV.

The IPv4 Backup ERO sub-TLV describes a path segment using IPv4 Address style of encoding. Its semantics have been borrowed from [RFC3209].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Flags         |                     Reserved                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     IPv4 Address (4 octets)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv4 Backup ERO sub-TLV format

where:

Type: TBD, suggested value 10

Length: 8 bytes

Flags: 1 octet field of following flags:

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

where:

L-bit - If the L bit is set, then the value of the attribute is 'loose.' Otherwise, the value of the attribute is 'strict.'

IPv4 Address - the address of the explicit route hop.

4.2.2.5. IPv6 Backup ERO sub-TLV

IPv6 ERO sub-TLV is a sub-TLV of the SID/Label Binding sub-TLV.

The IPv6 Backup ERO sub-TLV describes a Backup path segment using IPv6 Address style of encoding. Its appearance and semantics have been borrowed from [RFC3209].

The 'L' bit in the Flags is a one-bit attribute. If the L bit is set, then the value of the attribute is 'loose.' Otherwise, the value of the attribute is 'strict.'

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Flags      |                   Reserved                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-                                                             -+
|                                                               |
+-                          IPv6 Address                       -+
|                                                               |
+-                                                             -+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv6 Backup ERO sub-TLV format

where:

Type: TBD, suggested value 11

Length: 8 bytes

Flags: 1 octet field of following flags:

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

where:

L-bit - If the L bit is set, then the value of the attribute is ’loose.’ Otherwise, the value of the attribute is ’strict.’

IPv6 Address - the address of the explicit route hop.

4.2.2.6. Unnumbered Interface ID Backup ERO sub-TLV

Unnumbered Interface ID Backup sub-TLV is a sub-TLV of the SID/Label Binding sub-TLV.

The appearance and semantics of the ’Unnumbered Interface ID’ have been borrowed from [RFC3477].

The Unnumbered Interface-ID ERO sub-TLV describes a path segment that spans over an unnumbered interface. Unnumbered interfaces are

referenced using the interface index. Interface indices are assigned local to the router and therefore not unique within a domain. All elements in an ERO path need to be unique within a domain and hence need to be disambiguated using a domain unique Router-ID.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Type           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Flags    |                    Reserved                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Router ID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Interface ID                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Unnumbered Interface ID Backup ERO sub-TLV format

where:

Type: TBD, suggested value 12

Length: 12 bytes

Flags: 1 octet field of following flags:

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

where:

L-bit - If the L bit is set, then the value of the attribute is 'loose.' Otherwise, the value of the attribute is 'strict.'

Router-ID: Router-ID of the next-hop.

Interface ID: is the identifier assigned to the link by the router specified by the Router-ID.

5. Adjacency Segment Identifier (Adj-SID)

An Adjacency Segment Identifier (Adj-SID) represents a router adjacency in Segment Routing. At the current stage of Segment Routing architecture it is assumed that the Adj-SID value has local significance (to the router).

5.1. Adj-SID sub-TLV

A new extended OSPFv3 LSAs, as defined in [I-D.ietf-ospf-ospfv3-lsa-extend], are used to advertise prefix SID in OSPFv3

Adj-SID sub-TLV is an optional sub-TLV of the Router-Link TLV as defined in [I-D.ietf-ospf-ospfv3-lsa-extend]. It MAY appear multiple times in Router-Link TLV. Examples where more than one Adj-SID may be used per neighbor are described in [I-D.filsfils-rtgwg-segment-routing-use-cases]. The structure of the Adj-SID Sub-TLV is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Type            |              Length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Flags         |     Weight    |             Reserved          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   SID/Label/Index (variable)                  |
+---------------------------------------------------------------+
```

where:

Type: TBD, suggested value 10.

Length: variable.

Flags. 1 octet field of following flags:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|B|V|L|S|       |
+-+-+-+-+-+-+-+-+
```

where:

B-Flag: Backup-flag: set if the Adj-SID refer to an adjacency being protected (e.g.: using IPFRR or MPLS-FRR) as described in [I-D.filsfils-rtgwg-segment-routing-use-cases].

The V-Flag: Value/Index Flag. If set, then the Prefix-SID carries an absolute value. If not set, then the Prefix-SID carries an index.

The L-Flag: Local/Global Flag. If set, then the value/index carried by the PrefixSID has local significance. If not set,

then the value/index carried by this subTLV has global significance.

The S-Flag. Set Flag. When set, the S-Flag indicates that the Adj-SID refers to a set of adjacencies (and therefore MAY be assigned to other adjacencies as well).

Other bits: MUST be zero when originated and ignored when received.

Weight: weight used for load-balancing purposes. The use of the weight is defined in [I-D.filsfils-rtgwg-segment-routing].

SID/Index/Label: label or index value depending on the V-bit setting.

Examples:

A 32 bit global index defining the offset in the SID/Label space advertised by this router - in this case the V and L flags MUST be unset.

A 24 bit local label where the 20 rightmost bits are used for encoding the label value - in this case the V and L flags MUST be set.

16 octet IPv6 address - in this case the V-flag MUST be set. The L-flag MUST be set for link-local IPv6 address and MUST be unset for IPv6 global unicast address.

A SR capable router MAY allocate an Adj-SID for each of its adjacencies and set the B-Flag when the adjacency is protected by a FRR mechanism (IP or MPLS) as described in [I-D.filsfils-rtgwg-segment-routing-use-cases].

5.2. LAN Adj-SID Sub-TLV

LAN Adj-SID is an optional sub-TLV of the Router-Link TLV. It MAY appear multiple times in Router-Link TLV. It is used to advertise SID/Label for adjacency to non-DR node on broadcast or NBMA network.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |     Weight    |             Reserved          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Neighbor ID                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     SID/Label/Index (variable)                |
+---------------------------------------------------------------+
```

where:

Type: TBD, suggested value 11.

Length: variable.

Flags. 1 octet field of following flags:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|B|V|L|S|       |
+-+-+-+-+-+-+-+-+
```

where:

B-Flag: Backup-flag: set if the LAN-Adj-SID refer to an adjacency being protected (e.g.: using IPFRR or MPLS-FRR) as described in [I-D.filsfils-rtgwg-segment-routing-use-cases].

The V-Flag: Value/Index Flag. If set, then the Prefix-SID carries an absolute value. If not set, then the Prefix-SID carries an index.

The L-Flag: Local/Global Flag. If set, then the value/index carried by the PrefixSID has local significance. If not set, then the value/index carried by this subTLV has global significance.

The S-Flag. Set Flag. When set, the S-Flag indicates that the Adj-SID refers to a set of adjacencies (and therefore MAY be assigned to other adjacencies as well).

Other bits: MUST be zero when originated and ignored when received.

Weight: weight used for load-balancing purposes. The use of the weight is defined in [I-D.filsfils-rtgwg-segment-routing].

SID/Index/Label: label or index value depending on the V-bit setting.

Examples:

A 32 bit global index defining the offset in the SID/Label space advertised by this router - in this case the V and L flags MUST be unset.

A 24 bit local label where the 20 rightmost bits are used for encoding the label value - in this case the V and L flags MUST be set.

16 octet IPv6 address - in this case the V-flag MUST be set. The L-flag MUST be set for link-local IPv6 address and MUST be unset for IPv6 global unicast address.

6. Elements of Procedure

6.1. Intra-area Segment routing in OSPFv3

The OSPFv3 node that supports segment routing MAY advertise Prefix-SIDs for any prefix that it is advertising reachability for (e.g. loopback IP address) as described in Section 4.1.

If multiple routers advertise Prefix-SID for the same prefix, then the Prefix-SID MUST be the same. This is required in order to allow traffic load-balancing if multiple equal cost paths to the destination exist in the network.

Prefix-SID can also be advertised by the SR Mapping Servers (as described in [I-D.filsfils-rtgwg-segment-routing-use-cases]). The Mapping Server advertises Prefix-SID for remote prefixes that exist in the network. Multiple Mapping Servers can advertise Prefix-SID for the same prefix, in which case the same Prefix-SID MUST be advertised by all of them. SR Mapping Server could use either area scope or autonomous system flooding scope when advertising Prefix SID for prefixes, based on the configuration of the SR Mapping Server. Depending on the flooding scope used, SR Mapping Server chooses the LSA that will be used. If the area flooding scope is needed, E-Intra-Area-Prefix-LSA ([I-D.ietf-ospf-ospfv3-lsa-extend]) is used. If autonomous system flooding scope is needed, E-AS-External-LSA ([I-D.ietf-ospf-ospfv3-lsa-extend]) is used.

When Prefix-SID is advertised by the Mapping Server, which is indicated by the M-flag in the Prefix-SID sub-TLV (Section 4.1), route-type as indicated by the LSA type which is being used for flooding is ignored. Prefix SID is bound to a prefix, in which case route-type becomes unimportant.

Advertisement of the Prefix-SID by the Mapping Server using Inter-Area Prefix TLV, External Prefix TLV or Intra-Area-Prefix TLV ([I-D.ietf-ospf-ospfv3-lsa-extend]) does not itself contribute to the prefix reachability. NU-bit MUST be set in the PrefixOptions field of the LSA which is used by the Mapping Server to advertise SID or SID range, which prevents such advertisement to contribute to the prefix reachability.

6.2. Inter-area Segment routing in OSPFv3

In order to support SR in a multi-area environment, OSPFv3 must propagate Prefix-SID information between areas. The following procedure is used in order to propagate Prefix SIDs between areas.

When an OSPFv3 ABR advertises a Inter-Area-Prefix-LSA from an intra-area prefix to all its connected areas, it will also include Prefix-SID sub-TLV, as described in Section 4.1. The Prefix-SID value will be set as follows:

The ABR will look at its best path to the prefix in the source area and find out the advertising router associated with its best path to that prefix.

If no Prefix-SID was advertised for the prefix in the source area by the router that contributes to the best path to the prefix, then the ABR will use the Prefix-SID advertised by any other router (e.g.: a Prefix-SID coming from an SR Mapping Server as defined in [I-D.filsfils-rtgwg-segment-routing-use-cases]) when propagating Prefix-SID for the prefix to other areas.

When an OSPFv3 ABR advertises Inter-Area-Prefix-LSA LSAs from an inter-area route to all its connected areas it will also include Prefix-SID sub-TLV, as described in Section 4.1. The Prefix-SID value will be set as follows:

The ABR will look at its best path to the prefix in the source area and find out the advertising router associated with its best path to that prefix.

The ABR will then look if such router advertised a Prefix-SID for the prefix and use it when advertising the Prefix-SID to other connected areas.

If no Prefix-SID was advertised for the prefix in the source area by the ABR that contributes to the best path to the prefix, the originating ABR will use the Prefix-SID advertised by any other router (e.g.: a Prefix-SID coming from an SR Mapping Server as defined in [I-D.filsfils-rtgwg-segment-routing-use-cases]) when propagating Prefix-SID for the prefix to other areas.

6.3. SID for External Prefixes

AS-External-LSAs are flooded domain wide. When an ASBR, which supports SR, generates AS-External-LSA, it should also include Prefix-SID sub-TLV, as described in Section 4.1 Prefix-SID value will be set to the SID that has been reserved for that prefix.

When a NSSA ASBR translates NSSA-LSA into AS-External-LSA, it should also advertise the Prefix-SID for the prefix. The NSSA ABR determines its best path to the prefix advertised in the translated NSSA-LSA and finds the advertising router associated with such path. If such advertising router has advertised a Prefix-SID for the prefix, then the NSSA ASBR uses it when advertising the Prefix-SID in AS-External-LSA. Otherwise the Prefix-SID advertised by any other router will be used (e.g.: a Prefix-SID coming from an SR Mapping Server as defined in [I-D.filsfils-rtgwg-segment-routing-use-cases]).

6.4. Advertisement of Adj-SID

The Adjacency Segment Routing Identifier (Adj-SID) is advertised using the Adj-SID Sub-TLV as described in Section 5.

6.4.1. Advertisement of Adj-SID on Point-to-Point Links

Adj-SID MAY be advertised for any adjacency on p2p link that is in a state 2-Way or higher. If the adjacency on a p2p link transitions from the FULL state, then the Adj-SID for that adjacency MAY be removed from the area. If the adjacency transitions to a state lower then 2-Way, then the Adj-SID MUST be removed from the area.

6.4.2. Adjacency SID on Broadcast or NBMA Interfaces

Broadcast or NBMA networks in OSPFv3 are represented by a star topology where the Designated Router (DR) is the central point all other routers on the broadcast or NBMA network connect to. As a result, routers on the broadcast or NBMA network advertise only their adjacency to DR and BDR. Routers that are neither DR nor BDR do not form and do not advertise adjacencies between them. They, however, maintain a 2-Way adjacency state between them.

When Segment Routing is used, each router on the broadcast or NBMA network MAY advertise the Adj-SID for its adjacency to DR using Adj-SID Sub-TLV as described in Section 5.1.

SR capable router MAY also advertise Adj-SID for other neighbors (e.g. BDR, DR-OTHER) on broadcast or NBMA network using the LAN ADJ-SID Sub-TLV as described in section 5.1.1.2. Section 5.2.

7. IANA Considerations

This specification updates two existing OSPF registries.

7.1. OSPF Router Information (RI) TLVs Registry

- o suggested value 8 - SR-Algorithm TLV
- o suggested value 9 - SID/Label Range TLV

7.2. OSPFv3 Extend-LSA sub-TLV registry

- o suggested value 1 - SID/Label sub-TLV
- o suggested value 2 - Prefix SID sub-TLV
- o suggested value 3 - Adj-SID sub-TLV
- o suggested value 4 - LAN Adj-SID sub-TLV
- o suggested value 5 - SID/Label Binding sub-TLV
- o suggested value 6 - ERO Metric sub-TLV
- o suggested value 7 - IPv4 ERO sub-TLV
- o suggested value 8 - IPv6 ERO sub-TLV
- o suggested value 9 - Unnumbered Interface ID ERO sub-TLV
- o suggested value 10 - IPv4 Backup ERO sub-TLV
- o suggested value 11 - IPv6 Backup ERO sub-TLV
- o suggested value 12 - Unnumbered Interface ID Backup ERO sub-TLV

8. Security Considerations

Implementations must assure that malformed permutations of the newly defined sub-TLvs do not result in errors which cause hard OSPFv3 failures.

9. Contributors

The following people gave a substantial contribution to the content of this document: Ahmed Bashandy, Martin Horneffer, Bruno Decraene, Stephane Litkowski, Igor Milojevic, Rob Shakir and Saku Ytti.

10. Acknowledgements

We would like to thank Anton Smirnov for his contribution.

Many thanks to Yakov Rekhter, John Drake and Shraddha Hedge for their contribution on earlier incarnations of the "Binding / MPLS Label TLV" in [I-D.gredler-ospf-label-advertisement].

11. References

11.1. Normative References

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

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3477] Kompella, K. and Y. Rekhter, "Signalling Unnumbered Links in Resource ReSerVation Protocol - Traffic Engineering (RSVP-TE)", RFC 3477, January 2003.

[RFC4970] Lindem, A., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 4970, July 2007.

11.2. Informative References

[I-D.filsfils-rtgwg-segment-routing] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-rtgwg-segment-routing-01 (work in progress), October 2013.

[I-D.filsfils-rtgwg-segment-routing-use-cases]
Filsfils, C., Francois, P., Previdi, S., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., Kini, S., and E. Crabbe, "Segment Routing Use Cases", draft-filsfils-rtgwg-segment-routing-use-cases-02 (work in progress), October 2013.

[I-D.gredler-ospf-label-advertisement]
Gredler, H., Amante, S., Scholl, T., and L. Jalil, "Advertising MPLS labels in OSPF", draft-gredler-ospf-label-advertisement-03 (work in progress), May 2013.

[I-D.ietf-ospf-ospfv3-lsa-extend]
Lindem, A., Mirtorabi, S., Roy, A., and F. Baker, "OSPFv3 LSA Extendibility", draft-ietf-ospf-ospfv3-lsa-extend-03 (work in progress), May 2014.

[I-D.minto-rsvp-lsp-egress-fast-protection]
Jeganathan, J., Gredler, H., and Y. Shen, "RSVP-TE LSP egress fast-protection", draft-minto-rsvp-lsp-egress-fast-protection-03 (work in progress), November 2013.

Authors' Addresses

Peter Psenak (editor)
Cisco Systems, Inc.
Apollo Business Center
Mlynske nivy 43
Bratislava 821 09
Slovakia

Email: ppsenak@cisco.com

Stefano Previdi (editor)
Cisco Systems, Inc.
Via Del Serafico, 200
Rome 00142
Italy

Email: sprevidi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Brussels
Belgium

Email: cfilsfil@cisco.com

Hannes Gredler
Juniper Networks, Inc.
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
US

Email: hannes@juniper.net

Rob Shakir
British Telecom
London
UK

Email: rob.shakir@bt.com

Wim Henderickx
Alcatel-Lucent
Copernicuslaan 50
Antwerp 2018
BE

Email: wim.henderickx@alcatel-lucent.com

Jeff Tantsura
Ericsson
300 Holger Way
San Jose, CA 95134
US

Email: Jeff.Tantsura@ericsson.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 21, 2015

X. Xu
Huawei
U. Chunduri
Ericsson
M. Bhatia
Alcatel-Lucent

September 17, 2014

Carrying Routable IP Addresses in OSPF RI LSA
draft-xu-ospf-routable-ip-address-01

Abstract

This document proposes two new TLVs within the body of the OSPF Router Information (RI) Opaque LSA, called Routable IPv4 Address TLV and Routable IPv6 Address TLV. Here the OSPF means both OSPFv2 and OSPFv3.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

There are several situations where it is required for OSPF routers in one area to find correlations between routable IP addresses and capabilities of OSPF routers in another area. One example is the Entropy Label Capability (ELC) advertisement [I-D.xu-ospf-mpls-elc] across the OSPF domain. In this example, assume the ELC TLV originated by a router in one area is propagated to another area, those routers in the latter area need to find routable IP addresses of the router originating that ELC TLV before inserting the Entropy Label (EL) for packets going to the Label Switch Path (LSP) tunnel towards one of the above routable IP addresses. Another example is the S-BFD discriminator distribution [I-D.bhatia-ospf-sbfd-discriminator] across the OSPF domain. In this example, assume the S-BFD Discriminator TLV originated by a router in one area is propagated to another area, those routers in the latter area need to find routable IP addresses of the router originating that S-BFD Discriminator TLV so as to set up S-BDF sessions with that originating router.

However, in the OSPF Router Information (RI) Opaque LSA as defined in [RFC4970], which is used by OSPF routers to announce their capabilities, there is no such field for containing routable IP addresses of the originating router. Therefore, this document propose two new TLVs within the body of OSPF RI LSA, called Routable IPv4 Address TLV and Routable IPv6 Address TLV, which are used to carry routable IPv4 and IPv6 addresses respectively. Here the OSPF means both OSPFv2 and OSPFv3.

1.1. 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].

2. Terminology

This memo makes use of the terms defined in [RFC4970].

3. Routable IPv4 Address TLV

A new TLV within the body of the OSPF RI Opaque LSA, called Routable IPv4 Address TLV is defined to carry one or more routable IPv4 addresses of the router originating the RI LSA. The Type of this TLV is TBD, the Length is variable (multiple of 4), and the Value field contains one or more routable IPv4 addresses of the router originating the RI LSA. This TLV is applicable to OSPFv2 and for IPv4 Address Families (AFs) of OSPFv3 [RFC5838] within the body of the corresponding RI Opaque LSA. The scope of the advertisement MUST be domain-wide.

4. Routable IPv6 Address TLV

A new TLV within the body of the OSPFv3 RI Opaque LSA, called Routable IPv6 Address TLV is defined to carry one or more routable IPv6 global addresses of the router originating the RI LSA. The Type of this TLV is TBD, the Length is variable (multiple of 16), and the Value field contains one or more routable IPv6 global addresses of the router originating the RI LSA. This TLV is only applicable to OSPFv3. The scope of the advertisement MUST be domain-wide.

5. Acknowledgements

Thanks Karsten Thomann, Anton Smirnov, Joel Jaeggli, Joel M. Halpern, Wes George, Les Ginsberg, Abhay Roy, Mach Chen and Acee Lindem for their support and valuable comments on this draft.

6. IANA Considerations

This document includes a request to IANA to allocate two TLV type codes for the new RI LSA TLVs proposed in this document respectively.

7. Security Considerations

This document does not introduce any new security risk.

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.

[RFC4970] Lindem, A., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 4970, July 2007.

8.2. Informative References

[I-D.bhatia-ospf-sbfd-discriminator]
Bhatia, M., Ranganath, T., Pignataro, C., and S. Aldrin, "OSPF extensions to advertise S-BFD Target Discriminator", draft-bhatia-ospf-sbfd-discriminator-00 (work in progress), May 2014.

[I-D.xu-ospf-mpls-elc]
Xu, X., Kini, S., Sivabalan, S., and C. Filsfils, "Signaling Entropy Label Capability Using OSPF", draft-xu-ospf-mpls-elc-00 (work in progress), December 2013.

[RFC5838] Lindem, A., Mirtorabi, S., Roy, A., Barnes, M., and R. Aggarwal, "Support of Address Families in OSPFv3", RFC 5838, April 2010.

Authors' Addresses

Xiaohu Xu
Huawei

Email: xuxiaohu@huawei.com

Uma Chunduri
Ericsson

Email: uma.chunduri@ericsson.com

Manav Bhatia
Alcatel-Lucent

Email: manav.bhatia@alcatel-lucent.com

Internet
Internet-Draft
Intended status: Informational
Expires: April 17, 2015

D. Yeung
Y. Qu
Cisco Systems
J. Zhang
D. Bogdanovic
Juniper Networks
K. Sreenivasa
Brocade Communications System
October 14, 2014

Yang Data Model for OSPF Protocol
draft-yeung-netmod-ospf-02

Abstract

This document defines a YANG data model that can be used to configure and manage OSPF.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 17, 2015.

Table of Contents

1. Overview

YANG [RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces (e.g. ReST) and encodings other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interfaces, such as CLI and programmatic APIs.

A core routing data model is defined in [I-D.ietf-netmod-routing-cfg], and it proposes a basis for the development of data models for routing protocols. The interface data model is defined in [RFC7223] and is used for referencing interface from the routing protocol. This document defines a YANG data model that can be used to configure and manage OSPF and it is an augment to the core routing data model.

This document defines a YANG data model that can be used to configure and manage OSPF. Both OSPFv2 [RFC2328] and OSPFv3 [RFC5340] are supported. In additional to the core OSPF protocol, features described in different separate OSPF RFCs are also supported. They includes demand circuit [RFC1793], traffic engineering [RFC3630],

multiple address family [RFC5838], graceful restart [RFC3623] [RFC5187], NSSA [RFC3101] and sham link [RFC4577]. Those non-core features are made optional in the data model provided.

1.1. 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].

2. Design of Data Model

Although the basis of OSPF configuration elements like routers, areas and interfaces remains the same, the detailed configuration model varies among different vendors. Differences are observed in term of how protocol engine is tied to routing domain, how multiple protocol engines could be instantiated and configuration inheritance, just to name a few.

The goal of this document is to define a data model that is capable of representing these differences. There is very little information that is designated as "mandatory", providing freedom to vendors to adapt this data model to their product implementation.

2.1. Overview

The OSPF YANG module defined in this document has all the common building blocks for OSPF protocol.

The OSPF YANG module augments the routing/routing-instance/routing-protocols/routing-protocol path of the ietf-routing module.

```
module: ospf
+--rw routing
 +--rw routing-instance [name]
  +--rw routing-protocols
    +--rw routing-protocol [name]
       +--rw ospf
             .
             .
          +--rw all-instances-inherit {instance-inheritance}?
          |   .
          |   .
          +--rw instance* [routing-instance af]
                .
                .
             +--rw all-areas-inherit {area-inheritance}?
             |  .
             |  .
             +--rw area* [area-id]
             |  .
             |  .
             |  +--rw all-interfaces-inherit {interface-inheritance}?
             |  .
             |  .
             |  +--rw interface* [interface]
             |  .
             |  .
             +--rw topology* [name]
```

The ospf is intended to match to the vendor specific OSPF configuration construct which is identified by a local identifier 'name'. The field 'version' allows support for OSPFv2 and OSPFv3.

The ospf container includes one or more OSPF protocol engines, each encapsulated in the instance entity. Each instance includes information for the routing domain it is running on based on the [routing-instance af] specification. There is no default routing domain assumed by the data model. For example, to enable OSPF on the default IPv4 routing domain of the vendor, this model requires an explicit instance entity with the specification like ["default" "ipv4-unicast"]. The instance also contains OSPF router level configuration

The instance/area and instance/area/interface container contain the OSPF configuration for the area and interface level respectively

The instance/topology container contain the OSPF configuration for topology when multi-topology feature is enabled

2.2. OSPFv2 and OSPFv3

The defined data model supports both OSPFv2 and OSPFv3.

The field ’version’ is used to indicate the OSPF version and is a mandatory. Based on the version set, the data model change accordingly to accommodate the difference between the two versions.

2.3. Optional Features

Optional features are features beyond the basic of OSPF configurations and it is up to a vendor to decide the support of a particular feature on a particular device.

This module has declared a number of features, such as NSR, max-LSA etc.. It is intended that vendors will extend the features list.

2.4. Inheritance

This defined data model supports configuration inheritance for intances, areas and interfaces.

The all-instances-inherit, all-areas-inherit and all-interfaces-inherit containers provides a consistent way to configure inheritable command. Inheritance is treated as a feature. Vendors are expected to augment the above container to provide the list of inheritance command for their implementation.

2.5. OSPF Router Configuration

The container ospf is the top level container in this data model. It contains shared information among different OSPF instances under the container.

```
module: ospf
   +--rw ospf
      +--rw all-instances-inherit {instance-inheritance}?
      |  +--rw area
      |  +--rw interface
      +--rw operation-mode?          identityref
      +--rw instance* [routing-instance af]
         .
         .
```

2.6. OSPF Instance Configuration

The container instance represents an OSPF protocol engine. Each instance indicates the routing domain it is associated with based on [routing-instance af] and contains the router level configurations.

The all-areas-inherit container contains area configuration that could be inherited to all OSPF areas defined. Similarly, the all-areas-inherit also contains interface configuration that could be inherited to all the OSPF interfaces defined.

```
module: ospf

   +--rw ospf
         .
         .
      +--rw instance* [routing-instance af]
         +--rw routing-instance     rt:routing-instance-ref
         +--rw af                   identityref
         +--rw router-id?           yang:dotted-quad {router-id}?
         +--rw admin-distance
         |  +--rw (granularity)?
         |  |  +--:(detail)
         |  |  |  +--rw intra-area?   uint8
         |  |  |  +--rw inter-area?   uint8
         |  |  +--:(coarse)
         |  |     +--rw internal?     uint8
         |  +--rw external?     uint8
         +--rw nsr {nsr}?
         |  +--rw enable?   boolean
         +--rw graceful-restart {graceful-restart}?
         |  +--rw enable?                      boolean
         |  +--rw helper-enable?               boolean
         |  +--rw restart-interval?            uint16
         |  +--rw helper-strict-lsa-checking?   boolean
         +--rw protocol-shutdown {protocol-shutdown}?
         |  +--rw shutdown?   boolean
         +--rw auto-cost {auto-cost}?
         |  +--rw enable?                boolean
         |  +--rw reference-bandwidth?   uint32
         +--rw maximum
         |  +--rw paths?     uint16 {max-ecmp}?
         |  +--rw max-lsa?   uint32 {max-lsa}?
         +--rw mpls
         |  +--rw te-rid {te-rid}?
         |  |  +--rw (source)?
         |  |     +--:(interface)
         |  |     |  +--rw interface?   if:interface-ref
         |  |     +--:(explicit)
         |  |        +--rw router-id?   inet:ipv4-address
         |  +--rw ldp
         |     +--rw igp-sync?     boolean {ldp-igp-sync}?
         |     +--rw autoconfig?   boolean {ldp-igp-autoconfig}?
         +--rw all-areas-inherit {area-inheritance}?
         |  +--rw area
         |  +--rw interface
```

2.7. OSPF Area Configuration

The container area contains configurations of that area and the list of interface container represents all the OSPF interfaces active in the enclosing area.

```
module: ospf
 +--rw ospf
      .
      .
    +--rw instance* [routing-instance af]
        .
        .
      +--rw area* [area-id]
      |  +--rw area-id                   area-id-type
      |  +--rw area-type?                identityref
      |  +--rw summary?                  boolean
      |  +--rw default-cost?             uint32
      |  +--rw virtual-link* [router-id]
      |  |  +--rw router-id              yang:dotted-quad
      |  |  +--rw cost?                  uint16
      |  |  +--rw hello-interval?        uint16
      |  |  +--rw dead-interval?         uint16
      |  |  +--rw retransmit-interval?   uint16
      |  |  +--rw transmit-delay?        uint16
      |  |  +--rw mtu-ignore?            boolean {mtu-ignore}?
      |  |  +--rw lls?                   boolean {lls}?
      |  |  +--rw prefix-suppression?    boolean {prefix-suppression}?
      |  |  +--rw bfd?                   boolean {bfd}?
      |  |  +--rw ttl-security {ttl-security}?
      |  |  |  +--rw enable?   boolean
      |  |  |  +--rw hops?     uint8
      |  |  +--rw protocol-shutdown {protocol-if-shutdown}?
      |  |     +--rw shutdown?   boolean
      |  +--rw sham-link* [local-id remote-id]
      |  |  +--rw local-id               inet:ip-address
      |  |  +--rw remote-id              inet:ip-address
      |  |  +--rw cost?                  uint16
      |  |  +--rw hello-interval?        uint16
      |  |  +--rw dead-interval?         uint16
      |  |  +--rw retransmit-interval?   uint16
      |  |  +--rw transmit-delay?        uint16
      |  |  +--rw mtu-ignore?            boolean {mtu-ignore}?
      |  |  +--rw lls?                   boolean {lls}?
      |  |  +--rw prefix-suppression?    boolean {prefix-suppression}?
      |  |  +--rw bfd?                   boolean {bfd}?
      |  |  +--rw ttl-security {ttl-security}?
      |  |  |  +--rw enable?   boolean
```

```
        |  |  |  +--rw hops?     uint8
        |  |  +--rw protocol-shutdown {protocol-if-shutdown}?
        |  |     +--rw shutdown?   boolean
        |  +--rw range* [prefix]
        |  |  +--rw prefix       inet:ip-prefix
        |  |  +--rw advertise?   boolean
        |  |  +--rw cost?        uint24
        |  +--rw all-interfaces-inherit {interface-inheritance}?
        |  |  +--rw interface
```

2.8. OSPF Interface Configuration

The container interface contains configurations of that interface.

The ospf-interfaces also contain interface configuration that could be inherited to all ospf-interface’s defined.

```
module: ospf
 +--rw ospf
      .
      .
   +--rw instance* [routing-instance af]
       .
       .
     +--rw area* [area-id]
         .
         .
     |  +--rw interface* [interface]
     |     +--rw interface              if:interface-ref
     |     +--rw network-type?          enumeration
     |     +--rw passive?               boolean
     |     +--rw demand-circuit?        boolean {demand-circuit}?
     |     +--rw multi-area {multi-area-adj}?
     |     |  +--rw multi-area-id?   area-id-type
     |     |  +--rw cost?            uint16
     |     +--rw static-neighbors
     |     |  +--rw neighbor* [address]
     |     |     +--rw address          inet:ip-address
     |     |     +--rw cost?            uint16
     |     |     +--rw poll-interval?   uint16
     |     |     +--rw priority?        uint8
     |     +--rw cost?                  uint16
     |     +--rw hello-interval?        uint16
     |     +--rw dead-interval?         uint16
     |     +--rw retransmit-interval?   uint16
     |     +--rw transmit-delay?        uint16
     |     +--rw mtu-ignore?            boolean {mtu-ignore}?
     |     +--rw lls?                   boolean {lls}?
     |     +--rw prefix-suppression?    boolean {prefix-suppression}?
     |     +--rw bfd?                   boolean {bfd}?
     |     +--rw ttl-security {ttl-security}?
     |     |  +--rw enable?   boolean
     |     |  +--rw hops?     uint8
     |     +--rw protocol-shutdown {protocol-if-shutdown}?
     |     |  +--rw shutdown?   boolean
     |     +--rw topology* [name]
     |        +--rw name    rt:rib-ref
     |        +--rw cost?   uint32
```

2.9. OSPF notification

This YANG model defines a list of notifications to inform client of important events detected during the protocol operation. The notifications defined cover the common set of traps from OSPFv2 MIB [RFC4750] and OSPFv3 MIB [RFC5643].

```
module: ospf
notifications:
   +---n if-state-change
      |  +--ro routing-instance?        rt:routing-instance-ref
      |  +--ro routing-protocol-name?   string
      |  +--ro instance-af
      |  |  +--ro af?   identityref
      |  +--ro link-type?               identityref
      |  +--ro interface
      |  |  +--ro interface?   if:interface-ref
      |  +--ro virtual-link
      |  |  +--ro area-id?              uint32
      |  |  +--ro neighbor-router-id?   yang:dotted-quad
      |  +--ro sham-link
      |  |  +--ro area-id?          uint32
      |  |  +--ro local-ip-addr?    inet:ip-address
      |  |  +--ro remote-ip-addr?   inet:ip-address
      |  +--ro state?                   if-state-type
   +---n if-config-error
      |  +--ro routing-instance?        rt:routing-instance-ref
      |  +--ro routing-protocol-name?   string
      |  +--ro instance-af
      |  |  +--ro af?   identityref
      |  +--ro link-type?               identityref
      |  +--ro interface
      |  |  +--ro interface?       if:interface-ref
      |  |  +--ro packet-source?   yang:dotted-quad
      |  +--ro virtual-link
      |  |  +--ro area-id?              uint32
      |  |  +--ro neighbor-router-id?   yang:dotted-quad
      |  +--ro sham-link
      |  |  +--ro area-id?          uint32
      |  |  +--ro local-ip-addr?    inet:ip-address
      |  |  +--ro remote-ip-addr?   inet:ip-address
      |  +--ro packet-type?             packet-type
      |  +--ro error?                   enumeration
   +---n nbr-state-change
      |  +--ro routing-instance?        rt:routing-instance-ref
      |  +--ro routing-protocol-name?   string
      |  +--ro instance-af
      |  |  +--ro af?   identityref
      |  +--ro link-type?               identityref
      |  +--ro interface
      |  |  +--ro interface?            if:interface-ref
      |  |  +--ro neighbor-router-id?   yang:dotted-quad
      |  |  +--ro neighbor-ip-addr?     yang:dotted-quad
      |  +--ro virtual-link
      |  |  +--ro area-id?              uint32
```

```
      |  |  +--ro neighbor-router-id?   yang:dotted-quad
      |  +--ro sham-link
      |  |  +--ro area-id?              uint32
      |  |  +--ro local-ip-addr?        inet:ip-address
      |  |  +--ro neighbor-router-id?   yang:dotted-quad
      |  |  +--ro neighbor-ip-addr?     yang:dotted-quad
      |  +--ro state?                   nbr-state-type
      +---n nbr-restart-helper-status-change
      |  +--ro routing-instance?        rt:routing-instance-ref
      |  +--ro routing-protocol-name?   string
      |  +--ro instance-af
      |  |  +--ro af?   identityref
      |  +--ro link-type?               identityref
      |  +--ro interface
      |  |  +--ro interface?            if:interface-ref
      |  |  +--ro neighbor-router-id?   yang:dotted-quad
      |  |  +--ro neighbor-ip-addr?     yang:dotted-quad
      |  +--ro virtual-link
      |  |  +--ro area-id?              uint32
      |  |  +--ro neighbor-router-id?   yang:dotted-quad
      |  +--ro status?                  restart-helper-status-type
      |  +--ro age?                     uint32
      |  +--ro exit-reason?             restart-exit-reason-type
      +---n rx-bad-packet
      |  +--ro routing-instance?        rt:routing-instance-ref
      |  +--ro routing-protocol-name?   string
      |  +--ro instance-af
      |  |  +--ro af?   identityref
      |  +--ro link-type?               identityref
      |  +--ro interface
      |  |  +--ro interface?       if:interface-ref
      |  |  +--ro packet-source?   yang:dotted-quad
      |  +--ro virtual-link
      |  |  +--ro area-id?              uint32
      |  |  +--ro neighbor-router-id?   yang:dotted-quad
      |  +--ro sham-link
      |  |  +--ro area-id?          uint32
      |  |  +--ro local-ip-addr?    inet:ip-address
      |  |  +--ro remote-ip-addr?   inet:ip-address
      |  +--ro packet-type?             packet-type
      +---n lsdb-approaching-overflow
      |  +--ro routing-instance?        rt:routing-instance-ref
      |  +--ro routing-protocol-name?   string
      |  +--ro instance-af
      |  |  +--ro af?   identityref
      |  +--ro ext-lsdb-limit?          uint32
      +---n lsdb-overflow
      |  +--ro routing-instance?        rt:routing-instance-ref
```

```
      |  +--ro routing-protocol-name?   string
      |  +--ro instance-af
      |  |  +--ro af?   identityref
      |  +--ro ext-lsdb-limit?          uint32
      +---n nssa-translator-status-change
      |  +--ro routing-instance?        rt:routing-instance-ref
      |  +--ro routing-protocol-name?   string
      |  +--ro instance-af
      |  |  +--ro af?   identityref
      |  +--ro area-id?                 uint32
      |  +--ro status?                  nssa-translator-state-type
      +---n restart-status-change
         +--ro routing-instance?        rt:routing-instance-ref
         +--ro routing-protocol-name?   string
         +--ro instance-af
         |  +--ro af?   identityref
         +--ro status?                  restart-status-type
         +--ro restart-interval?        uint16
         +--ro exit-reason?             restart-exit-reason-type
```

3. OSPF Yang Module

```
<CODE BEGINS>
module ospf {
  namespace "urn:ietf:params:xml:ns:yang:ospf";
  // replace with IANA namespace when assigned
  prefix ospf;

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

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

  import ietf-interfaces {
    prefix "if";
  }

  import ietf-routing {
    prefix "rt";
  }

  organization
    "Cisco Systems
     170 West Tasman Drive
     San Jose, CA 95134-1706
```

```
        USA";
     contact
       "Derek Yeung myeung@cisco.com
        Yingzhen Qu yiqu@cisco.com
        Dean Bogdanovic deanb@juniper.net
        Jeffrey Zhang zzhang@juniper.net
        Kiran Agrahara Sreenivasa kkoushik@Brocade.com";

     description
       "This YANG module defines the generic configuration
        data for OSPF, which is common across all of the vendor
        implementations of the protocol. It is intended that the module
        will be extended by vendors to define vendor-specific
        OSPF configuration parameters and policies,
        for example route maps or route policies.


        Terms and Acronyms

        OSPF (ospf): Open Shortest Path First

        IP (ip): Internet Protocol

        IPv4 (ipv4):Internet Protocol Version 4

        IPv6 (ipv6): Internet Protocol Version 6

        MTU (mtu) Maximum Transmission Unit
       ";

     revision 2014-09-17 {
       description
         "Initial revision.";
       reference
         "RFC XXXX: A YANG Data Model for OSPF";
     }

     identity ospfv2 {
       base "rt:routing-protocol";
       description "OSPFv2";
     }

     identity ospfv3 {
       base "rt:routing-protocol";
       description "OSPFv3";
     }

     identity operation-mode {
```

```
      description
        "OSPF operation mode.";
    }

    identity ships-in-the-night {
      base operation-mode;
      description
        "Ships-in-the-night operation mode in which
        each OSPF instance carries only one address family";
    }

    identity area-type {
      description "Base identity for OSPF area type.";
    }

    identity normal {
      base area-type;
      description "OSPF normal area.";
    }

    identity stub {
      base area-type;
      description "OSPF stub area.";
    }

    typedef uint24 {
      type uint32 {
        range "0 .. 16777215";
      }
      description
        "24-bit unsigned integer.";
    }

    typedef area-id-type {
      type union {
        type uint32;
        type yang:dotted-quad;
      }
      description
        "Area ID type.";
    }

    typedef if-state-type {
      type enumeration {
        enum Down {
          value "1";
          description
            "Interface down state";
```

```
        }
        enum Loopback {
          value "2";
          description
            "Interface loopback state";
        }
        enum Waiting {
          value "3";
          description
            "Interface waiting state";
        }
        enum Point-to-Point {
          value "4";
          description
            "Interface point-to-point state";
        }
        enum DR {
          value "5";
          description
            "Interface Designated Router (DR) state";
        }
        enum BDR {
          value "6";
          description
            "Interface Backup Designated Router (BDR) state";
        }
        enum DR-Other {
          value "7";
          description
            "Interface Other Designated Router state";
        }
      }
      description
        "OSPF interface state type.";
    }

    typedef nbr-state-type {
      type enumeration {
        enum Down {
          value "1";
          description
            "Neighbor down state";
        }
        enum Attempt {
          value "2";
          description
            "Neighbor attempt state";
        }
```

```
        enum Init {
          value "3";
          description
            "Neighbor init state";
        }
        enum 2-Way {
          value "4";
          description
            "Neighbor 2-Way state";
        }
        enum ExStart {
          value "5";
          description
            "Neighbor exchange start state";
        }
        enum Exchange {
          value "6";
          description
            "Neighbor exchange state";
        }
        enum Loading {
          value "7";
          description
            "Neighbor loading state";
        }
        enum Full {
          value "8";
          description
            "Neighbor full state";
        }
      }
      description
        "OSPF neighbor state type.";
    }

    typedef restart-helper-status-type {
      type enumeration {
        enum Not-Helping {
          value "1";
          description
            "Restart helper status not helping.";
        }
        enum Helping {
          value "2";
          description
            "Restart helper status helping.";
        }
      }
```

```
      description
        "Restart helper status type.";
    }

    typedef restart-exit-reason-type {
      type enumeration {
        enum None {
          value "1";
          description
            "Not attempted.";
        }
        enum InProgress {
          value "2";
          description
            "Restart in progress.";
        }
        enum Completed {
          value "3";
          description
            "Successfully completed.";
        }
        enum TimedOut {
          value "4";
          description
            "Timed out.";
        }
        enum TopologyChanged {
          value "5";
          description
            "Aborted due to topology change.";
        }
      }
      description
        "Describes the outcome of the last attempt at a
        graceful restart, either by itself or acting
        as a helper.";
    }

    typedef packet-type {
      type enumeration {
        enum Hello {
          value "1";
          description
            "OSPF hello packet.";
        }
        enum Database-Descripton {
          value "2";
          description
```

```
            "OSPF database description packet.";
        }
        enum Link-State-Request {
          value "3";
          description
            "OSPF link state request packet.";
        }
        enum Link-State-Update {
          value "4";
          description
            "OSPF link state update packet.";
        }
        enum Link-State-Ack {
          value "5";
          description
            "OSPF link state acknowlegement packet.";
        }
      }
      description
        "OSPF packet type.";
    }

    typedef nssa-translator-state-type {
      type enumeration {
        enum Enabled {
          value "1";
          description
            "NSSA translator enabled state.";
        }
        enum Elected {
          description
            "NSSA translator elected state.";
        }
        enum Disabled {
          value "3";
          description
            "NSSA translator disabled state.";
        }
      }
      description
        "OSPF NSSA translator state type.";
    }

    typedef restart-status-type {
      type enumeration {
        enum Not-Restarting {
          value "1";
          description
```

```
            "Router is not restarting.";
        }
        enum Planned-Restart {
          description
            "Router is going through planned restart.";
        }
        enum Unplanned-Restart {
          value "3";
          description
            "Router is going through unplanned restart.";
        }
      }
      description
        "OSPF graceful restart status type.";
    }

    feature multi-topology {
      description
        "Support MTR.";
    }

    feature multi-area-adj {
      description
        "OSPF multi-area adjacency support as in RFC 5185.";
    }

    feature router-id {
      description
        "Set router ID per instance.";
    }

    feature demand-circuit {
      description
        "OSPF demand circuit support as in RFC 1793.";
    }

    feature mtu-ignore {
      description
        "Disable OSPF MTU mismatch detection on receiving
         DBD packets.";
    }

    feature lls {
      description
        "OSPF link-local signaling (LLS) as in RFC 5613.";
    }

    feature prefix-suppression {
```

```
      description
        "OSPF prefix suppression support as in RFC 6860.";
    }

    feature bfd {
      description
        "OSPF BFD support.";
    }

    feature ttl-security {
      description
        "OSPF ttl security check.";
    }

    feature nsr {
      description
        "Non-Stop-Routing (NSR).";
    }

    feature graceful-restart {
      description
        "Graceful OSPF Restart as defined in RFC3623 and RFC5187.";
    }

    feature protocol-shutdown {
      description
        "Shutdown the protocol.";
    }

    feature auto-cost {
      description
        "Calculate OSPF interface cost according to
          reference bandwidth.";
    }

    feature max-ecmp {
      description
        "Setting maximum number of ECMP paths.";
    }

    feature max-lsa {
      description
        "Setting maximum number of LSAs OSPF will receive.";
    }

    feature te-rid {
      description
        "TE router-id.";
```

```
     }

     feature ldp-igp-sync {
       description
         "LDP IGP synchronization.";
     }

     feature ldp-igp-autoconfig {
       description
         "LDP IGP auto-config.";
     }

     feature protocol-if-shutdown {
       description
         "Shutdown the protocol over an interface.";
     }

     feature instance-inheritance {
       description
         "Support inheritance";
     }

     feature af-inheritance {
       description
         "Support inheritance";
     }

     feature area-inheritance {
       description
         "Support area inheritance";
     }

     feature interface-inheritance {
       description
         "Support interface inheritance";
     }

     grouping interface-common-config {
       description "Common configuration for all types of interfaces,
                    including virtual link and sham link";

       leaf cost {
         type uint16 {
           range "1..65535";
         }
         description
           "Interface cost.";
       }
```

```
        leaf hello-interval {
          type uint16 {
            range "1..65535";
          }
          units seconds;
          description
            "Time between hello packets.";
        }

        leaf dead-interval {
          type uint16 {
            range "1..65535";
          }
          units seconds;
          must "dead-interval > ../hello-interval" {
            error-message "The dead interval must be "
                        + "larger than the hello interval";
            description
              "The value MUST be greater than 'hello-internval'.";
          }
          description
            "Interval after which a neighbor is declared dead.";
        }

        leaf retransmit-interval {
          type uint16 {
            range "1..65535";
          }
          units seconds;
          description
            "Time between retransmitting unacknowledged Link State
             Advertisements (LSAs).";
        }

        leaf transmit-delay {
          type uint16 {
            range "1..65535";
          }
          units seconds;
          description
            "Estimated time needed to send link-state update.";
        }

        leaf mtu-ignore {
          if-feature mtu-ignore;
          type boolean;
          description
            "Enable/Disable ignoring of MTU in DBD packets.";
```

```
        }

        leaf lls {
          if-feature lls;
          type boolean;
          description
            "Enable/Disable link-local signaling (LLS) support.";
        }

        leaf prefix-suppression {
          if-feature prefix-suppression;
          type boolean;
          description
            "Suppress advertisement of the prefixes.";
        }

        leaf bfd {
          if-feature bfd;
          type boolean;
          description
            "Enable/disable bfd.";
        }

        container ttl-security {
          if-feature ttl-security;
          description "TTL security check.";
          leaf enable {
            type boolean;
            description
              "Enable/Disable TTL security check.";
          }
          leaf hops {
            type uint8 {
              range "1..254";
            }
            description
              "Maximum number of hops that a OSPF packet may
               have traveled.";
          }
        }
        container protocol-shutdown {
          if-feature protocol-if-shutdown;
          description
            "Protocol shutdown interface config state.";
          leaf shutdown {
            type boolean;
            description
              "Enable/Disable protocol shutdown on the interface.";
```

```
          }
        }
      } // interface-common-config

      grouping interface-config {
        description "Configuration for real interfaces.";

        leaf network-type {
          type enumeration {
            enum "broadcast" {
              description
                "Specify OSPF broadcast multi-access network.";
            }
            enum "non-broadcast" {
              description
                "Specify OSPF Non-Broadcast Multi-Access
                (NBMA) network.";
            }
            enum "point-to-multipoint" {
              description
                "Specify OSPF point-to-multipoint network.";
            }
            enum "point-to-point" {
              description
                "Specify OSPF point-to-point network.";
            }
          }
          description
            "Network type.";
        }

        leaf passive {
          type boolean;
          description
            "Enable/Disable passive.";
        }

        leaf demand-circuit {
          if-feature demand-circuit;
          type boolean;
          description
            "Enable/Disable demand circuit.";
        }

        container multi-area {
          if-feature multi-area-adj;
          description
            "Configure ospf multi-area.";
```

```
          leaf multi-area-id {
            type area-id-type;
            description
              "Multi-area ID";
          }
          leaf cost {
            type uint16;
            description
              "Interface cost for multi-area.";
          }
        }

        container static-neighbors {
          description "Static configured neighbors.";

          list neighbor {
            key "address";
            description
              "Specify a neighbor router.";

            leaf address {
              type inet:ip-address;
              description "Neighbor IP address.";
            }

            leaf cost {
              type uint16 {
                range "1..65535";
              }
              description "Neighbor cost.";
            }
            leaf poll-interval {
              type uint16 {
                range "1..65535";
              }
              units seconds;
              description "Neighbor poll interval.";
            }
            leaf priority {
              type uint8 {
                range "1..255";
              }
              description "Neighbor priority for DR election.";
            }
          }
        }

        uses interface-common-config;
```

```
     } // grouping interface-config

     grouping tlv {
       description
         "TLV";
       leaf type {
         type uint16;
         description "TLV type.";
       }
       leaf length {
         type uint16;
         description "TLV length.";
       }
       leaf value {
         type yang:hex-string;
         description "TLV value.";
       }
     }

     grouping ospfv2-lsa-body {
       description "OSPFv2 LSA body.";
       container router {
         when "../../header/type = 1" {
           description
             "Only apply to Router-LSA.";
         }
         description
           "Router LSA.";
         leaf flags {
           type bits {
             bit V {
               description
                 "When set, the router is an endpoint of one or
                 more virtual links.";
             }
             bit E {
               description
                 "When set, the router is an AS Boundary Router
                 (ASBR).";
             }
             bit B {
               description
                 "When set, the router is an Area Border Router (ABR).";
             }
           }
           description "Flags";
         }
         leaf num-of-links {
```

```
          type uint16;
          description "Number of links.";
        }
        list link {
          key "link-id link-data";
          description "Router LSA link.";
          leaf link-id {
            type union {
              type inet:ipv4-address;
              type yang:dotted-quad;
            }
            description "Link ID";
          }
          leaf link-data {
            type union {
              type inet:ipv4-address;
              type uint32;
            }
            description "Link data.";
          }
          leaf type {
            type uint8;
            description "Link type.";
          }
          list topology {
            key "mt-id";
            description
              "Topology specific information.";
            leaf mt-id {
              type uint8;
              description
                "The MT-ID for topology enabled on the link.";
            }
            leaf metric {
              type uint16;
              description "Metric for the topology.";
            }
          }
        }
      }
      container network {
        when "../../header/type = 2" {
          description
            "Only apply to network LSA.";
        }
        description
          "Network LSA.";
        leaf network-mask {
```

```
          type inet:ipv4-address;
          description
            "The IP address mask for the network";
        }
        leaf-list attached-router {
          type yang:dotted-quad;
          description
            "List of the routers attached to the network.";
        }
      }
      container summary {
        when "../../header/type = 3 or "
           + "../../header/type = 4" {
          description
            "Only apply to Summary-LSA.";
        }
        description
          "Summary LSA.";
        leaf network-mask {
          type inet:ipv4-address;
          description
            "The IP address mask for the network";
        }
        list topology {
          key "mt-id";
          description
            "Topology specific information.";
          leaf mt-id {
            type uint8;
            description
              "The MT-ID for topology enabled on the link.";
          }
          leaf metric {
            type uint24;
            description "Metric for the topology.";
          }
        }
      }
      container external {
        when "../../header/type = 5 or "
           + "../../header/type = 7" {
          description
            "Only apply to AS-external-LSA and NSSA-LSA.";
        }
        description
          "External LSA.";
        leaf network-mask {
          type inet:ipv4-address;
```

```
          description
            "The IP address mask for the network";
        }
        list topology {
          key "mt-id";
          description
            "Topology specific information.";
          leaf mt-id {
            type uint8;
            description
              "The MT-ID for topology enabled on the link.";
          }
          leaf flags {
            type bits {
              bit E {
                description
                  "When set, the metric specified is a Type 2
                  external metric.";
              }
            }
            description "Flags.";
          }
          leaf metric {
            type uint24;
            description "Metric for the topology.";
          }
          leaf forwarding-address {
            type inet:ipv4-address;
            description
              "Forwarding address.";
          }
          leaf external-route-tag {
            type uint32;
            description
              "Route tag.";
          }
        }
      }
      container opaque {
        when "../../header/type = 9 or "
           + "../../header/type = 10 or "
           + "../../header/type = 11" {
          description
            "Only apply to opaque LSA.";
        }
        description
          "Opaque LSA.";
```

```
         list unknown-tlv {
           key "type";
           description "Unknown TLV.";
           uses tlv;
         }

         container router-address-tlv {
           leaf router-address {
             type inet:ipv4-address;
             description
               "Router address.";
           }
           description
             "Router address TLV.";
         }

         container link-tlv {
           leaf link-type {
             type uint8;
             mandatory true;
             description "Link type.";
           }
           leaf link-id {
             type union {
               type inet:ipv4-address;
               type yang:dotted-quad;
             }
             mandatory true;
             description "Link ID.";
           }
           leaf-list local-if-ipv4-addr {
             type inet:ipv4-address;
             description
               "List of local interface IPv4 addresses.";
           }
           leaf-list local-remote-ipv4-addr {
             type inet:ipv4-address;
             description
               "List of remote interface IPv4 addresses.";
           }
           leaf te-metric {
             type uint32;
             description "TE metric.";
           }
           leaf max-bandwidth {
             type decimal64 {
               fraction-digits 2;
             }
```

```
            description "Maximum bandwidth.";
          }
          leaf max-reservable-bandwidth {
            type decimal64 {
              fraction-digits 2;
            }
            description "Maximum reservable bandwidth.";
          }
          leaf unreserved-bandwidth {
            type decimal64 {
              fraction-digits 2;
            }
            description "Unreserved bandwidth.";
          }
          leaf admin-group {
            type uint32;
            description "Administrative group/Resource class/Color.";
          }
          list unknown-subtlv {
            key "type";
            description "Unknown sub-TLV.";
            uses tlv;
          }
          description
            "Link TLV.";
        }
      }
    }

    grouping ospfv3-lsa-options {
      description "OSPFv3 LSA options";
      leaf options {
        type bits {
          bit DC {
            description
              "When set, the router support demand circuits.";
          }
          bit R {
            description
              "When set, the originator is an active router.";
          }
          bit N {
            description
              "If set, the router is attached to an NSSA";
          }
          bit E {
            description
              "This bit describes the way AS-external-LSAs
```

```
                are flooded";
            }
            bit V6 {
              description
                "If clear, the router/link should be excluded
                 from IPv6 routing calculaton";
            }
          }
          mandatory true;
          description "OSPFv3 LSA options.";
        }
      }

      grouping ospfv3-lsa-prefix {
        description
          "OSPFv3 LSA prefix.";

        leaf prefix {
          type inet:ip-prefix;
          description
            "Prefix";
        }
        leaf prefix-options {
          type bits {
            bit NU {
              description
                "When set, the prefix should be excluded
                from IPv6 unicast calculations.";
            }
            bit LA {
              description
                "When set, the prefix is actually an IPv6 interface
                address of the Advertising Router.";
            }
            bit P {
              description
                "When set, the NSSA area prefix should be
                readvertised by the translating NSSA area border.";
            }
            bit DN {
              description
                "When set, the inter-area-prefix-LSA or
                AS-external-LSA prefix has been advertised in a VPN
                environment.";
            }
          }
          mandatory true;
          description "Prefix options.";
```

```
      }
    }

    grouping ospfv3-lsa-external {
      description
        "AS-External and NSSA LSA.";
      leaf metric {
        type uint24;
        description "Metric";
      }

      leaf flags {
        type bits {
          bit E {
            description
              "When set, the metric specified is a Type 2
              external metric.";
          }
        }
        description "Flags.";
      }

      leaf referenced-ls-type {
        type uint16;
        description "Referenced Link State type.";
      }

      uses ospfv3-lsa-prefix;

      leaf forwarding-address {
        type inet:ipv6-address;
        description
          "Forwarding address.";
      }

      leaf external-route-tag {
        type uint32;
        description
          "Route tag.";
      }
      leaf referenced-link-state-id {
        type uint32;
        description
          "Referenced Link State ID.";
      }
    }

    grouping ospfv3-lsa-body {
```

```
        description "OSPFv3 LSA body.";
        container router {
          when "../../header/type = 8193" { // 0x2001
            description
              "Only apply to Router-LSA.";
          }
          description "Router LSA.";
          leaf flags {
            type bits {
              bit V {
                description
                  "When set, the router is an endpoint of one or
                  more virtual links.";
              }
              bit E {
                description
                  "When set, the router is an AS Boundary Router
                  (ASBR).";
              }
              bit B {
                description
                  "When set, the router is an Area Border Router (ABR).";
              }
              bit Nt {
                description
                  "When set, the router is an NSSA border router
                  that is unconditionally translating NSSA-LSAs
                  into AS-external-LSAs.";
              }
            }
            mandatory true;
            description "LSA option.";
          }

          uses ospfv3-lsa-options;

          list link {
            key "interface-id neighbor-interface-id neighbor-router-id";
            description "Router LSA link.";
            leaf interface-id {
              type uint32;
              description "Interface ID.";
            }
            leaf neighbor-interface-id {
              type uint32;
              description "Neighbor Interface ID.";
            }
            leaf neighbor-router-id {
```

```
            type yang:dotted-quad;
            description "Neighbor Router ID";
          }
          leaf type {
            type uint8;
            description "Link type.";
          }
          leaf metric {
            type uint16;
              description "Metric.";
          }
        }
      }
      container network {
        when "../../header/type = 8194" { // 0x2002
          description
            "Only apply to network LSA.";
        }
        description "Network LSA.";

        uses ospfv3-lsa-options;

        leaf-list attached-router {
          type yang:dotted-quad;
          description
            "List of the routers attached to the network.";
        }
      }
      container inter-area-prefix {
        when "../../header/type = 8195" { // 0x2003
          description
            "Only apply to inter-area-prefix LSA.";
        }
        leaf metric {
          type uint24;
          description "Metric";
        }

        uses ospfv3-lsa-prefix;
        description "Inter-Area-Prefix LSA.";
      }
      container inter-area-router {
        when "../../header/type = 8196" { // 0x2004
          description
            "Only apply to inter-area-router LSA.";
        }
        uses ospfv3-lsa-options;
        leaf metric {
```

```
          type uint24;
          description "Metric";
        }
        leaf destination-router-id {
          type yang:dotted-quad;
          description
            "The Router ID of the router being described by the LSA.";
        }
        description "Inter-Area-Router LSA.";
      }
      container as-external {
        when "../../header/type = 16389" { // 0x2007
          description
            "Only apply to as-external LSA.";
        }

        uses ospfv3-lsa-external;

        description "AS-External LSA.";
      }
      container nssa {
        when "../../header/type = 8199" { // 0x2007
          description
            "Only apply to nssa LSA.";
        }
        uses ospfv3-lsa-external;

        description "NSSA LSA.";
      }
      container link {
        when "../../header/type = 8" { // 0x0008
          description
            "Only apply to link LSA.";
        }
        leaf rtr-priority {
          type uint8;
          description "Router Priority of the interface.";
        }

        uses ospfv3-lsa-options;

        leaf link-local-interface-address {
          type inet:ipv6-address;
          description
            "The originating router's link-local
            interface address on the link.";
        }
```

```
          leaf num-of-prefixes {
            type uint32;
            description "Number of prefixes.";
          }

          list prefix {
            key "prefix";
            description "List of prefixes associated with the link.";
            uses ospfv3-lsa-prefix;
          }
          description "Link LSA.";
        }
        container intra-area-prefix {
          when "../../header/type = 8201" { // 0x2009
            description
              "Only apply to intra-area-prefix LSA.";
          }
          description "Intra-Area-Prefix LSA.";

          leaf referenced-ls-type {
            type uint16;
            description "Referenced Link State type.";
          }
          leaf referenced-link-state-id {
            type uint32;
            description
              "Referenced Link State ID.";
          }
          leaf referenced-adv-router {
            type inet:ipv4-address;
            description
              "Referenced Advertising Router.";
          }

          leaf num-of-prefixes {
            type uint16;
            description "Number of prefixes.";
          }
          list prefix {
            key "prefix";
            description "List of prefixes associated with the link.";
            uses ospfv3-lsa-prefix;
            leaf metric {
              type uint24;
              description "Metric";
            }
          }
        }
```

```
     }

     grouping lsa-header {
       description
           "Common LSA for OSPFv2 and OSPFv3";
       leaf age {
         type uint16;
         mandatory true;
         description "LSA age.";
       }
       leaf type {
         type uint16;
         mandatory true;
         description "LSA type.";
       }
       leaf adv-router {
         type yang:dotted-quad;
         mandatory true;
         description "LSA advertising router.";
       }
       leaf seq-num {
         type uint32;
         mandatory true;
         description "LSA sequence number.";
       }
       leaf checksum {
         type uint16;
         mandatory true;
         description "LSA checksum.";
       }
       leaf length {
         type uint16;
         mandatory true;
         description "LSA length.";
       }
     }

     grouping ospfv2-lsa {
       description
           "OSPFv2 LSA.";
       container header {
         description
           "Decoded OSPFv2 LSA header data.";
         leaf option {
           type bits {
             bit DC {
               description
                 "When set, the router support demand circuits.";
```

```
            }
            bit P {
              description
                "Only used in type-7 LSA. When set, the NSSA
                border router should translate the type-7 LSA
                to type-5 LSA.";
            }
            bit MC {
              description
                "When set, the router support MOSPF.";
            }
            bit E {
              description
                "This bit describes the way AS-external-LSAs
                are flooded";
            }
          }
          mandatory true;
          description "LSA option.";
        }
        leaf lsa-id {
          type inet:ipv4-address;
          mandatory true;
          description "LSA ID.";
        }

        leaf opaque-type {
          when "../../header/type = 9 or "
             + "../../header/type = 10 or "
             + "../../header/type = 11" {
            description
              "Only apply to opaque LSA.";
          }
          type uint8;
          mandatory true;
          description "Opaque type.";
        }

        leaf opaque-id {
          when "../../header/type = 9 or "
             + "../../header/type = 10 or "
             + "../../header/type = 11" {
            description
              "Only apply to opaque LSA.";
          }
          type uint24;
          mandatory true;
          description "Opaque id.";
```

```
        }
        uses lsa-header;
      }
      container body {
        description
          "Decoded OSPFv2 LSA body data.";
        uses ospfv2-lsa-body;
      }
    }

    grouping ospfv3-lsa {
      description
          "Decoded OSPFv3 LSA.";
      container header {
        description
          "Decoded OSPFv3 LSA header data.";
        leaf lsa-id {
          type uint32;
          mandatory true;
          description "LSA ID.";
        }
        uses lsa-header;
      }
      container body {
        description
          "Decoded OSPF LSA body data.";
        uses ospfv3-lsa-body;
      }
    }

    grouping lsa {
      description
          "OSPF LSA.";
      leaf decoded-completed {
        type boolean;
        description
          "The OSPF LSA body is fully decoded.";
      }
      leaf raw-data {
        type yang:hex-string;
        description
          "The complete LSA in network byte
          order as received/sent over the wire.";
      }
      choice version {
        description
          "OSPFv2 or OSPFv3 LSA body.";
        container ospfv2 {
```

```
          when "../../../rt:type = 'ospfv2'" {
            description "Applied to OSPFv2 only";
          }
          description "OSPFv2 LSA";
          uses ospfv2-lsa;
        }
        container ospfv3 {
          when "../../../rt:type = 'ospfv3'" {
            description "Applied to OSPFv3 only";
          }
          description "OSPFv3 LSA";
          uses ospfv3-lsa;
        }
      }
    }

    grouping lsa-key {
      description
        "OSPF LSA key.";
      leaf lsa-id {
        type union {
          type inet:ipv4-address;
          type uint32;
        }
        description
          "LSA ID.";
      }
      leaf adv-router {
        type inet:ipv4-address;
        description
          "Advertising router.";
      }
    }

    grouping af-area-config {
      description
        "OSPF address-family specific area config state.";
      list range {
        key "prefix";
        description
          "Summarize routes matching address/mask (border
           routers only)";
        leaf prefix {
          type inet:ip-prefix;
          description
            "IPv4 or IPv6 prefix";
        }
        leaf advertise {
```

```
          type boolean;
          description
            "Advertise or hide.";
        }
        leaf cost {
          type uint24 {
            range "0..16777214";
          }
          description
            "Cost of summary route.";
        }
      }
    }

    grouping area-config {
      description
        "OSPF area config state.";
      leaf area-type {
        type identityref {
          base area-type;
        }
        default normal;
        description
          "Area type.";
      }

      leaf summary {
        when "area-type = 'stub' or area-type = 'nssa'" {
          description
            "Summary generation valid for stub/NSSA area.";
        }
        type boolean;
        description
          "Enable/Disable summary generation to the stub or
          NSSA area.";
      }

      leaf default-cost {
        when "area-type = 'stub' or area-type = 'nssa'" {
            description
              "Default cost for LSA advertised into stub or
              NSSA area.";
        }
        type uint32 {
          range "1..16777215";
        }
        description
          "Set the summary default-cost for a stub or NSSA area.";
```

```
        }

        list virtual-link {
          key "router-id";
          description
            "OSPF virtual link";
          leaf router-id {
            type yang:dotted-quad;
            description
              "Virtual link router ID.";
          }

          uses interface-common-config;
        }

        list sham-link {
          key "local-id remote-id";
          description
            "OSPF sham link";
          leaf local-id {
            type inet:ip-address;
            description
              "Address of the local end-point.";
          }
          leaf remote-id {
            type inet:ip-address;
            description
              "Address of the remote end-point.";
          }
          uses interface-common-config;
        }

        uses af-area-config {
          when "../../operation-mode = 'ospf:ships-in-the-night'" {
            description
              "Ships in the night configuration.";
          }
        }
      }

      grouping instance-config {
        description
          "OSPF instance config state.";
        leaf router-id {
          if-feature router-id;
          type yang:dotted-quad;
          description
            "Defined in RFC 2328. A 32-bit number
```

```
             that uniquely identifies the router.";
       }

       container admin-distance {
         description "Admin distance config state.";
         choice granularity {
           description
             "Options for expressing admin distance
              for intra-area and inter-area route";
           case detail {
             leaf intra-area {
               type uint8;
               description
                 "Admin distance for intra-area route.";
             }
             leaf inter-area {
               type uint8;
               description
                 "Admin distance for inter-area route.";
             }
           }
           case coarse {
             leaf internal {
               type uint8;
               description
                 "Admin distance for both intra-area and
                  inter-area route.";
             }
           }
         }
         leaf external {
           type uint8;
           description
             "Admin distance for both external route.";
         }
       }

       container nsr {
         if-feature nsr;
         description
           "NSR config state.";
         leaf enable {
           type boolean;
           description
             "Enable/Disable NSR.";
         }
       }
```

```
        container graceful-restart {
          if-feature graceful-restart;
          description
            "Graceful restart config state.";
          leaf enable {
            type boolean;
            description
              "Enable/Disable graceful restart as defined in RFC 3623.";
          }
          leaf helper-enable {
            type boolean;
            description
              "Enable RestartHelperSupport in RFC 3623 Section B.2.";
          }
          leaf restart-interval {
            type uint16 {
              range "1..1800";  // Range is defined in RFC 3623.
            }
            units seconds;
            default "120";  // Default is defined in RFC 3623.
            description
              "RestartInterval option in RFC 3623 Section B.1.";
          }
          leaf helper-strict-lsa-checking {
            type boolean;
            description
              "RestartHelperStrictLSAChecking option in RFC 3623
              Section B.2.";
          }
        }

        container protocol-shutdown {
          if-feature protocol-shutdown;
          description
            "Protocol shutdown config state.";
          leaf shutdown {
            type boolean;
            description
              "Enable/Disable protocol shutdown.";
          }
        }

        container auto-cost {
          if-feature auto-cost;
          description
            "Auto cost config state.";
          leaf enable {
            type boolean;
```

```
          description
            "Enable/Disable auto cost.";
        }
        leaf reference-bandwidth {
          type uint32 {
            range "1..4294967";
          }
          units Mbits;
          description
            "Configure reference bandwidth in term of Mbits";
        }
      }

      container maximum {
        description
          "OSPF limits settings.";
        leaf paths {
          if-feature max-ecmp;
          type uint16 {
            range "1..32";
          }
          description
            "Maximum number of ECMP paths.";
        }
        leaf max-lsa {
          if-feature max-lsa;
          type uint32 {
            range "1..4294967294";
          }
          description
            "Maximum number of LSAs OSPF will receive.";
        }
      }

      container mpls {
        description
          "OSPF MPLS config state.";
        container te-rid {
          if-feature te-rid;
          description
            "Traffic Engineering stable IP address for system.";
          choice source {
            description
              "Different options for specifying TE router ID.";
            case interface {
              leaf interface {
                type if:interface-ref;
                description
```

```
                    "Take the interface's IPv4 address as TE router ID.";
                }
              }
              case explicit {
                leaf router-id {
                  type inet:ipv4-address;
                  description
                    "Explicitly configure the TE router ID.";
                }
              }
            }
          }
          container ldp {
            description
              "OSPF MPLS LDP config state.";
            leaf igp-sync {
              if-feature ldp-igp-sync;
              type boolean;
              description
                "Enable LDP IGP synchronization.";
            }
            leaf autoconfig {
              if-feature ldp-igp-autoconfig;
              type boolean;
              description
                "Enable LDP IGP interface auto-configuration.";
            }
          }
        }
      }


      grouping interface-operation {
        description
          "OSPF interface operation state.";
        reference "RFC2328 Section 9";
        uses interface-config;

        leaf state {
          type if-state-type;
          description "Interface state.";
        }

        leaf hello-timer {
          type uint32;
          units "milliseconds";
          description "Hello timer.";
        }
```

```
        leaf wait-timer {
          type uint32;
          units "milliseconds";
          description "Wait timer.";
        }

        list neighbor {
          description
            "List of neighbors.";
          leaf neighbor-id {
            type leafref {
              path "../../neighbor/neighbor-id";
            }
            description "Neighbor.";
          }
        }

        leaf dr {
          type inet:ipv4-address;
          description "DR.";
        }

        leaf bdr {
          type inet:ipv4-address;
          description "BDR.";
        }
      } // interface-operation

      grouping neighbor-operation {
        description
          "OSPF neighbor operation data.";

        leaf address {
          type inet:ip-address;
          description
            "Neighbor address.";
        }
        leaf dr {
          type inet:ipv4-address;
          description
            "Designated Router.";
        }
        leaf bdr {
          type inet:ipv4-address;
          description
            "Backup Designated Router.";
        }
        leaf state {
```

```
        type nbr-state-type;
        description
          "OSPF neighbor state.";
      }
    }

    grouping instance-operation {
      description
        "OSPF Address Family operation state.";
      leaf router-id {
        type yang:dotted-quad;
        description
          "Defined in RFC 2328. A 32-bit number
           that uniquely identifies the router.";
      }
    }

    augment "/rt:routing/rt:routing-instance/rt:routing-protocols/"
          + "rt:routing-protocol" {
      when "rt:type = 'ospf:ospfv2' or rt:type = 'ospf:ospfv3'" {
        description
          "This augment is only valid for a routing protocol instance
           of OSPF (type 'ospfv2' or 'ospfv3').";
      }
      description "OSPF augmentation.";

      container ospf {
        description
          "OSPF.";

        container all-instances-inherit {
          if-feature instance-inheritance;
          description
            "Inheritance support to all instances.";
          container area {
            description
              "Area config to be inherited by all areas in
              all instances.";
          }
          container interface {
            description
              "Interface config to be inherited by all interfaces
              in all instances.";
          }
        }

        leaf operation-mode {
          type identityref {
```

```
            base operation-mode;
          }
          default ospf:ships-in-the-night;
          description
            "OSPF operation mode.";
        }

        list instance {
          key "routing-instance af";
          description
            "An OSPF routing protocol instance.";
          leaf routing-instance {
            type rt:routing-instance-ref;
            description
              "For protocol centric model, which is supported in
              default-instance only, this could reference any layer 3
              routing-instance.
              For routing-instance centric model, must reference the
              enclosing routing-instance.";
          }

          leaf af {
            type identityref {
              base rt:address-family;
            }
            description
              "Address-family of the instance.";
          }

          uses instance-config;

          container all-areas-inherit {
            if-feature area-inheritance;
            description
              "Inheritance for all areas.";
            container area {
              description
                "Area config to be inherited by all areas.";
            }
            container interface {
              description
                "Interface config to be inherited by all interfaces
                in all areas.";
            }
          }

          list area {
            key "area-id";
```

```
            description
              "List of ospf areas";
            leaf area-id {
              type area-id-type;
              description
                "Area ID.";
            }

            uses area-config;

            container all-interfaces-inherit {
              if-feature interface-inheritance;
              description
                "Inheritance for all interfaces";
              container interface {
                description
                  "Interface config to be inherited by all
                  interfaces.";
              }
            }

            list interface {
              key "interface";
              description
                "List of OSPF interfaces.";
              leaf interface {
                type if:interface-ref;
                description
                  "Interface.";
              }
              uses interface-config;
            } // list of interfaces
          } // list of areas
        } // list of instance
      } // container ospf
    }


    augment "/rt:routing/rt:routing-instance/rt:routing-protocols/"
          + "rt:routing-protocol/ospf:ospf/ospf:instance" {
      when "../rt:type = 'ospf:ospfv2' or ../rt:type = 'ospf:ospfv3'" {
        description
          "This augment is only valid for OSPF
          (type 'ospfv2' or 'ospfv3').";
      }
      if-feature multi-topology;
      description
        "OSPF multi-topology routing-protocol augmentation.";
```

```
        list topology {
          // Topology must be in the same routing-instance
          //  and of same AF as the container.
          key "name";
          description "OSPF topology.";
          leaf name {
            type rt:rib-ref;
            description "RIB";
          }
          list area {
            key "area-id";
            description
              "List of ospf areas";
            leaf area-id {
              type area-id-type;
              description
                "Area ID.";
            }
            uses area-config;
          }
        }
      }

      augment "/rt:routing/rt:routing-instance/rt:routing-protocols/"
            + "rt:routing-protocol/ospf:ospf/ospf:instance/"
            + "ospf:area/ospf:interface" {
        when "../../rt:type = 'ospf:ospfv2'" {
          description
            "This augment is only valid for OSPFv2.";
        }
        if-feature ospf:multi-topology;
        description "OSPF multi-topology interface augmentation.";
        list topology {
          key "name";
          description "OSPF interface topology.";
          leaf name {
            type rt:rib-ref;
            description
              "One of the topology enabled on this interface";
          }
          leaf cost {
             type uint32;
             description
               "Interface cost for this topology";
          }
        }
      }
```

```
     augment "/rt:routing-state/rt:routing-instance/"
           + "rt:routing-protocols/rt:routing-protocol" {
       when "rt:type = 'ospf:ospfv2' or rt:type = 'ospf:ospfv3'"  {
         description
           "This augment is only valid for a routing protocol instance
            of type 'ospfv2' or 'ospfv3'.";
       }
       description
            "OSPF configuration.";
       container ospf {
         description "OSPF";

         leaf operation-mode {
           type identityref {
             base operation-mode;
           }
           description
             "OSPF operation mode.";
         }

         list instance {
           key "routing-instance af";
           description
             "An OSPF routing protocol instance.";
           leaf routing-instance {
             type rt:routing-instance-ref;
             description
               "For protocol centric model, which is supported in
               default-instance only, this could reference any layer 3
               routing-instance.
               For routing-instance centric model, must reference the
               enclosing routing-instance.";
           }

           leaf af {
             type identityref {
               base rt:address-family;
             }
             description
               "Address-family of the instance.";
           }

           uses instance-operation;

           list neighbor {
             key "area-id interface neighbor-id";
             description
               "List of OSPF neighbors.";
```

```
              leaf area-id {
                type area-id-type;
                description
                  "Area ID.";
              }
              leaf interface {
                // Should it refer to config state leaf?
                type if:interface-ref;
                description
                  "Interface.";
              }
              leaf neighbor-id {
                type inet:ipv4-address;
                description
                  "Neighbor ID.";
              }

              uses neighbor-operation;
            } // list of OSPF neighbors

            list interface {
              key "area-id interface";
              description
                "List of OSPF interfaces.";

              leaf area-id {
                type area-id-type;
                description "Area ID.";
              }
              leaf interface {
                // Should it refer to config state leaf?
                type if:interface-ref;
                description "Interface.";
              }

              uses interface-operation;
            } // list of OSPF interfaces

            list area {
              key "area-id";
              description "List of OSPF areas";
              leaf area-id {
                type area-id-type;
                description "Area ID.";
              }
            } // list of OSPF areas

            container databases {
```

```
              description
                "OSPF databases.";
              list link-scope-lsas {
                when "../../../rt:type = 'ospfv3'" {
                  description
                    "Link scope LSA only exists in OSPFv3.";
                }
                key "area-id interface lsa-type";
                description "List OSPF link scope LSA databases";
                leaf area-id {
                  type uint32; // Should it refer to config state leaf?
                  description "Area ID.";
                }
                leaf interface {
                  // Should it refer to config state leaf?
                  type if:interface-ref;
                  description "Interface.";
                }
                leaf lsa-type {
                  type uint8;
                  description "OSPF link scope LSA type.";
                }
                list link-scope-lsa {
                  key "lsa-id adv-router";
                  description "List of OSPF link scope LSAs";
                  uses lsa-key;
                  uses lsa;
                }
              } // list link-scope-lsas

              list area-scope-lsas {
                key "area-id lsa-type";
                description "List OSPF area scope LSA databases";
                leaf lsa-type {
                  type uint8;
                  description "OSPF area scope LSA type.";
                }
                leaf area-id {
                  type uint32; // Should it refer to config state leaf?
                  description "Area ID.";
                }
                list area-scope-lsa {
                  key "lsa-id adv-router";
                  description "List of OSPF area scope LSAs";
                  uses lsa-key;
                  uses lsa;
                }
              } // list area-scope-lsas
```

```
            list as-scope-lsas {
              key "lsa-type";
              description "List OSPF AS scope LSA databases";
              leaf lsa-type {
                type uint8;
                description "OSPF AS scope LSA type.";
              }
              list as-scope-lsa {
                key "lsa-id adv-router";
                description "List of OSPF AS scope LSAs";
                uses lsa-key;
                uses lsa;
              }
            } // list as-scope-lsas
          } // container databases
        }
      } // container ospf
    }

    augment "/rt:routing-state/rt:routing-instance/"
          + "rt:routing-protocols/rt:routing-protocol/"
          + "ospf:ospf/ospf:instance" {
      when "../rt:type = 'ospf:ospfv2'" {
        description
          "This augment is only valid for OSPFv2.";
      }
      if-feature multi-topology;
      description
        "OSPF multi-topology routing-protocol augmentation.";
      list topology {
        // Topology must be in the same routing-instance
        // and of same AF as the container.
        key "name";
        description "OSPF topology.";
        leaf name {
          type rt:rib-ref;
          description "RIB";
        }
        list area {
          key "area-id";
          description
            "List of ospf areas";
          leaf area-id {
            type area-id-type;
            description
              "Area ID.";
          }
        }
```

```
      }
    }

    augment "/rt:routing-state/rt:routing-instance/"
          + "rt:routing-protocols/rt:routing-protocol/"
          + "ospf:ospf/ospf:instance/ospf:interface" {
      when "../../rt:type = 'ospf:ospfv2'" {
        description
          "This augment is only valid for OSPFv2.";
      }
      if-feature ospf:multi-topology;
      description "OSPF multi-topology interface augmentation.";
      list topology {
        key "name";
        description "OSPF interface topology.";
        leaf name {
          type rt:rib-ref;
          description
            "One of the topology enabled on this interface";
        }
      }
    }

    grouping route-content {
      description
        "This grouping defines OSPF-specific route attributes.";
      leaf metric {
        type uint32;
        description "OSPF route metric.";
      }
      leaf tag {
        type uint32;
        default "0";
        description "OSPF route tag.";
      }
      leaf route-type {
        type enumeration {
          enum intra-area {
            description "OSPF intra-area route";
          }
          enum inter-area {
            description "OSPF inter-area route";
          }
          enum external-1 {
            description "OSPF external route type 1";
          }
          enum external-2 {
            description "OSPF External route type 2";
```

```
            }
            enum nssa-1 {
              description "OSPF NSSA external route type 1";
            }
            enum nssa-2 {
              description "OSPF NSSA external route type 2";
            }
          }
          description "OSPF route type";
        }
      }

      augment "/rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route" {
        when "rt:source-protocol = 'ospf:ospfv2' or "
           + "rt:source-protocol = 'ospf:ospfv3'" {
          description
            "This augment is only valid for a routes whose source
             protocol is OSPF.";
          }
         description
           "OSPF-specific route attributes.";
         uses route-content;
      }

      augment "/rt:active-route/rt:output/rt:route" {
        description
          "OSPF-specific route attributes in the output of 'active-route'
          RPC.";
        uses route-content;
      }

      identity if-link-type {
        description "Base identity for OSPF interface link type.";
      }

      identity if-link-type-normal {
        base if-link-type;
        description "OSPF interface link type normal.";
      }

      identity if-link-type-virtual-link {
        base if-link-type;
        description "OSPF interface link type virtual link.";
      }

      identity if-link-type-sham-link {
        base if-link-type;
        description "OSPF interface link type sham link.";
```

```
     }

     grouping notification-instance-hdr {
       description
        "This group describes common instance specific
         data for notifications.";

       leaf routing-instance {
         type rt:routing-instance-ref;
         description
           "Describe the routing instance.";
       }

       leaf routing-protocol-name {
         type string;
         description
          "Describes the name of the OSPF routing protocol.";
       }

       container instance-af {
         leaf af {
           type identityref {
             base rt:address-family;
           }
           description
             "Address-family of the instance.";
         }
         description
          "Describes the address family of the OSPF instance.";
       }
     }

     notification if-state-change {
       uses notification-instance-hdr;

       leaf link-type {
         type identityref {
           base if-link-type;
         }
         description "Type of OSPF interface.";
       }

       container interface {
         description "Normal interface.";
         leaf interface {
           type if:interface-ref;
           description "Interface.";
         }
```

```
        }
        container virtual-link {
          description "virtual-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf neighbor-router-id {
            type yang:dotted-quad;
            description "Neighbor router id.";
          }
        }
        container sham-link {
          description "sham-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf local-ip-addr {
            type inet:ip-address;
            description "Sham link local address.";
          }

          leaf remote-ip-addr {
            type inet:ip-address;
            description "Sham link remote address.";
          }
        }

        leaf state {
          type if-state-type;
          description "Interface state.";
        }

        description
          "This notification is sent when interface
          state change is detected.";
      }

      notification if-config-error {
        uses notification-instance-hdr;

        leaf link-type {
          type identityref {
            base if-link-type;
          }
          description "Type of OSPF interface.";
        }
```

```
        container interface {
          description "Normal interface.";
          leaf interface {
            type if:interface-ref;
            description "Interface.";
          }
          leaf packet-source {
            type yang:dotted-quad;
            description "Source address.";
          }
        }
        container virtual-link {
          description "virtual-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf neighbor-router-id {
            type yang:dotted-quad;
            description "Neighbor router id.";
          }
        }

        container sham-link {
          description "sham-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf local-ip-addr {
            type inet:ip-address;
            description "Sham link local address.";
          }

          leaf remote-ip-addr {
            type inet:ip-address;
            description "Sham link remote address.";
          }
        }

        leaf packet-type {
          type packet-type;
          description "OSPF packet type.";
        }

        leaf error {
          type enumeration {
            enum "badVersion" {
```

```
            description "Bad version";
          }
          enum "areaMismatch" {
            description "Area mistmatch";
          }
          enum "unknownNbmaNbr" {
            description "Unknown NBMA neighbor";
          }
          enum "unknownVirtualNbr" {
            description "Unknown virtual link neighbor";
          }
          enum "authTypeMismatch" {
            description "Auth type mismatch";
          }
          enum "authFailure" {
            description "Auth failure";
          }
          enum "netMaskMismatch" {
            description "Network mask mismatch";
          }
          enum "helloIntervalMismatch" {
            description "Hello interval mismatch";
          }
          enum "deadIntervalMismatch" {
            description "Dead interval mismatch";
          }
          enum "optionMismatch" {
            description "Option mismatch";
          }
          enum "mtuMismatch" {
            description "MTU mismatch";
          }
          enum "duplicateRouterId" {
            description "Duplicate router ID";
          }
          enum "noError" {
            description "No error";
          }
        }
        description "Error code.";
      }
      description
        "This notification is sent when interface
        config error is detected.";
    }

    notification nbr-state-change {
      uses notification-instance-hdr;
```

```
        leaf link-type {
          type identityref {
            base if-link-type;
          }
          description "Type of OSPF interface.";
        }

        container interface {
          description "Normal interface.";
          leaf interface {
            type if:interface-ref;
            description "Interface.";
          }
          leaf neighbor-router-id {
            type yang:dotted-quad;
            description "Neighbor router id.";
          }
          leaf neighbor-ip-addr {
            type yang:dotted-quad;
            description "Neighbor address.";
          }
        }
        container virtual-link {
          description "virtual-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf neighbor-router-id {
            type yang:dotted-quad;
            description "Neighbor router id.";
          }
        }
        container sham-link {
          description "sham-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf local-ip-addr {
            type inet:ip-address;
            description "Sham link local address.";
          }
          leaf neighbor-router-id {
            type yang:dotted-quad;
            description "Neighbor router id.";
          }
          leaf neighbor-ip-addr {
```

```
            type yang:dotted-quad;
            description "Neighbor address.";
          }
        }

        leaf state {
            type nbr-state-type;
          description "Neighbor state.";
        }

        description
          "This notification is sent when neighbor
          state change is detected.";
      }

      notification nbr-restart-helper-status-change {
        uses notification-instance-hdr;

        leaf link-type {
          type identityref {
            base if-link-type;
          }
          description "Type of OSPF interface.";
        }

        container interface {
          description "Normal interface.";
          leaf interface {
            type if:interface-ref;
            description "Interface.";
          }
          leaf neighbor-router-id {
            type yang:dotted-quad;
            description "Neighbor router id.";
          }
          leaf neighbor-ip-addr {
            type yang:dotted-quad;
            description "Neighbor address.";
          }
        }
        container virtual-link {
          description "virtual-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf neighbor-router-id {
            type yang:dotted-quad;
```

```
            description "Neighbor router id.";
          }
        }

        leaf status {
          type restart-helper-status-type;
          description "Restart helper status.";
        }

        leaf age {
          type uint32;
          units seconds;
          description
            "Remaining time in current OSPF graceful restart
            interval, if the router is acting as a restart
            helper for the neighbor.";
        }

        leaf exit-reason {
          type restart-exit-reason-type;
          description
            "Restart helper exit reason.";
        }
        description
          "This notification is sent when neighbor restart
          helper status change is detected.";
      }

      notification rx-bad-packet {
        uses notification-instance-hdr;

        leaf link-type {
          type identityref {
            base if-link-type;
          }
          description "Type of OSPF interface.";
        }

        container interface {
          description "Normal interface.";
          leaf interface {
            type if:interface-ref;
            description "Interface.";
          }
          leaf packet-source {
            type yang:dotted-quad;
            description "Source address.";
          }
```

```
        }
        container virtual-link {
          description "virtual-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf neighbor-router-id {
            type yang:dotted-quad;
            description "Neighbor router id.";
          }
        }

        container sham-link {
          description "sham-link.";
          leaf area-id {
            type uint32;
            description "Area ID.";
          }
          leaf local-ip-addr {
            type inet:ip-address;
            description "Sham link local address.";
          }

          leaf remote-ip-addr {
            type inet:ip-address;
            description "Sham link remote address.";
          }
        }

        leaf packet-type {
          type packet-type;
          description "OSPF packet type.";
        }

        description
          "This notification is sent when an OSPF packet
          has been received on a interface that cannot be parsed.";
      }

      notification lsdb-approaching-overflow {
        uses notification-instance-hdr;

        leaf ext-lsdb-limit {
          type uint32;
          description
            "The maximum number of non-default AS-external LSAs
            entries that can be stored in the link state database.";
```

```
        }

        description
          "This notification is sent when the number of LSAs
          in the router's link state database has exceeded
          ninety percent of the ext-lsdb-limit.";
      }

      notification lsdb-overflow {
        uses notification-instance-hdr;

        leaf ext-lsdb-limit {
          type uint32;
          description
            "The maximum number of non-default AS-external LSAs
            entries that can be stored in the link state database.";
        }

        description
          "This notification is sent when the number of LSAs
          in the router's link state database has exceeded
          ext-lsdb-limit.";
      }

      notification nssa-translator-status-change {
        uses notification-instance-hdr;

        leaf area-id {
          type uint32;
          description "Area ID.";
        }

        leaf status {
          type nssa-translator-state-type;
          description
            "NSSA translator status.";
        }

        description
          "This notification is sent when there is a change
          in the router's ability to translate OSPF NSSA LSAs
          OSPF AS-External LSAs.";
      }

      notification restart-status-change {
        uses notification-instance-hdr;

        leaf status {
```

```
        type restart-status-type;
        description
          "Restart status.";
      }

      leaf restart-interval {
        type uint16 {
          range "1..1800";
        }
        units seconds;
        default "120";
        description
          "Restart interval.";
      }

      leaf exit-reason {
        type restart-exit-reason-type;
        description
          "Restart exit reason.";
      }

      description
        "This notification is sent when the graceful restart
        state for the router has changed.";
    }
  }
  <CODE ENDS>
```

4. Security Considerations

The data model defined does not create any security implications.

This draft does not change any underlying security issues inherent in [I-D.ietf-netmod-routing-cfg].

5. Acknowledgements

The authors wish to thank Acee Lindem, Yi Yang, Alexander Clemm, Gaurav Gupta, Ing-Wher Chen, Ladislav Lhotka and Stephane Litkowski for their thorough reviews and helpful comments.

This document was produced using Marshall Rose’s xml2rfc tool.

6. References

6.1. Normative References

[RFC1793] Moy, J., "Extending OSPF to Support Demand Circuits", RFC 1793, April 1995.

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC3101] Murphy, P., "The OSPF Not-So-Stubby Area (NSSA) Option", RFC 3101, January 2003.

[RFC3623] Moy, J., Pillay-Esnault, P., and A. Lindem, "Graceful OSPF Restart", RFC 3623, November 2003.

[RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, September 2003.

[RFC4577] Rosen, E., Psenak, P., and P. Pillay-Esnault, "OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4577, June 2006.

[RFC4750] Joyal, D., Galecki, P., Giacalone, S., Coltun, R., and F. Baker, "OSPF Version 2 Management Information Base", RFC 4750, December 2006.

[RFC5187] Pillay-Esnault, P. and A. Lindem, "OSPFv3 Graceful Restart", RFC 5187, June 2008.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

[RFC5643] Joyal, D. and V. Manral, "Management Information Base for OSPFv3", RFC 5643, August 2009.

[RFC5838] Lindem, A., Mirtorabi, S., Roy, A., Barnes, M., and R. Aggarwal, "Support of Address Families in OSPFv3", RFC 5838, April 2010.

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

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, May 2014.

6.2. Informative References

[I-D.ietf-netmod-routing-cfg] Lhotka, L., "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-15 (work in progress), May 2014.

Authors' Addresses

Derek Yeung
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134
USA

EMail: myeung@cisco.com

Yingzhen Qu
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134
USA

EMail: yiqu@cisco.com

Jeffrey Zhang
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

EMail: zzhang@juniper.net

Dean Bogdanovic
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

EMail: deanb@juniper.net

Kiran Agrahara Sreenivasa
Brocade Communications System
9442 Capital of Texas Hwy North
Arboretum Plaza One, Suite 500
Austin, TX 78759
USA

EMail: kkoushik@brocade.com

Open Shortest Path First
Internet-Draft
Updates: 2328, 5340 (if approved)
Intended status: Standards Track
Expires: April 17, 2015

Z. Zhang
L. Wang
Juniper Networks, Inc.
A. Lindem
Cisco Systems
D. Dubois
General Dynamics C4S
V. Julka
T. McMillan
L3 Communications, Linkabit

October 14, 2014

OSPF Two-part Metric
draft-zzhang-ospf-two-part-metric-05.txt

Abstract

This document specifies an optional extension to the OSPF protocol, to represent the metric on a multi-access network as two parts: the metric from a router to the network, and the metric from the network to the router. The router to router metric would be the sum of the two.

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

Copyright Notice

Table of Contents

1. Introduction

For a broadcast network, a Network-LSA is advertised to list all routers on the network, and each router on the network includes a link in its Router-LSA to describe its connection to the network. The link in the Router-LSA includes a metric but the listed routers in the Network LSA do not include a metric. This is based on the assumption that from a particular router, all others on the same network can be reached with the same metric.

With some broadcast networks, different routers can be reached with different metrics. RFC 6845 extends the OSPF protocol with a hybrid interface type for that kind of broadcast network, where no Network LSA is advertised and Router-LSAs simply include p2p links to all routers on the same network with individual metrics. Broadcast capability is still utilized to optimize database synchronization and adjacency maintenance.

That works well for broadcast networks where the metric between different pair of routers are really independent. For example, VPLS networks.

With certain types of broadcast networks, further optimization can be made to reduce the size of the Router-LSAs and number of updates.

Consider a satellite radio network with fixed and mobile ground terminals. All communication goes through the satellite. When the mobile terminals move about, their communication capability may change. When OSPF runs over the radio network (routers being or in tandem with the terminals), RFC 6845 hybrid interface can be used, but with the following drawbacks.

Consider that one terminal/router moves into an area where its communication capability degrades significantly. Through the radio control protocol, all other routers determine that the metric to this particular router changed and they all need to update their Router-LSAs accordingly. The router in question also determines that its metric to reach all others also changed and it also needs to update its Router-LSA. Consider that there could be many terminals and many of them can be moving fast and frequently, the number/frequency of updates of those large Router-LSAs could inhibit network scaling.

2. Proposed Enhancement

Notice that in the above scenario, when one terminal’s communication capability changes, its metric to all other terminals and the metric from all other terminals to it will all change in a similar fashion. Given this, the above problem can be easily addressed by breaking the

metric into two parts: the metric to the satellite and the metric from the satellite. The metric from terminal R1 to R2 would be the sum of the metric from R1 to the satellite and the metric from the satellite to R2.

Now instead of using the RFC 6845 hybrid interface type, the network is just treated as a regular broadcast network. A router on the network no longer lists individual metrics to each neighbor in its Router-LSA. Instead, each router advertises the metric from the network to itself in addition to the normal metric for the network. With the normal Router-to-Network and additional Network-to-Router metrics advertised for each router, individual router-to-router metric can be calculated.

With the proposed enhancement, the size of Router-LSA will be significantly reduced. In addition, when a router’s communication capability changes, only that router needs to update its Router-LSA.

Note that while the example uses the satellite as the relay point at the radio level (layer-2), at layer-3, the satellite does not participate in packet forwarding. In fact, the satellite does not need to be running any layer-3 protocol. Therefore for generality, the metric is abstracted as to/from the "network" rather that specifically to/from the "satellite".

3. Speficications

The following protocol specifications are added to or modified from the base OSPF protocol. If an area contains one or more two-part metric networks, then all routers in the area must support the extensions specified herein. This is ensured by procedures described in Section 3.5.

3.1. Router Interface Parameters

The "Router interface parameters" have the following additions:

- o Two-part metric: TRUE if the interface connects to a multi-access network that uses two-part metric. All routers connected to the same network SHOULD have the same configuration for their corresponding interfaces.

- o Interface input cost: Link state metric from the two-part-metric network to this router. Defaulted to "Interface output cost" but not valid for normal networks using a single metric. May be configured or dynamically adjusted to a value different from the "Interface output cost".

3.2. Advertising Network-to-Router metric in OSPFv2

For OSPFv2, the Network-to-Router metric is encoded in an OSPF Extended Link TLV Sub-TLV [ietf-ospf-lsa-extend], defined in this document as the Network-to-Router Metric Sub-TLV. The type of the Sub-TLV is TBD. The length of the Sub-TLV is 4 (for the value part only). The value part of the Sub-TLV 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      MT       |        0      |          MT   metric          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Multiple such Sub-TLVs can exist in a single OSPF Extended Link TLV, one for each topology. The OSPF Extended Link TLV identifies the transit link to the network, and is part of an OSPFv2 Extended-Link Opaque LSA. The Sub-TLV MUST ONLY appear in Extended-Link TLVs for Link Type 2 (link to transit network), and MUST be ignored if received for other link types.

3.3. Advertising Network-to-Router metric in OSPFv3

For OSPFv3, the same Network-to-Router Metric Sub-TLV definition is used, though it is part of the Router-Link TLV of E-Router-LSA [ietf-ospf-ospfv3-lsa-extend]. Currently OSPFv3 Multi-Toplogy is not defined so the only valid value for the MT field is 0 and only one such Sub-TLV SHOULD be included in the Router-Link TLV. Received Sub-TLVs with non-zero MT field MUST be ignored.

Similarly, the Sub-TLV MUST ONLY appear in Router-Link TLVs for Link Type 2 (connection to a transit network) and MUST be ignored if received for other link types.

3.4. SPF Calculation

During the first stage of shortest-path tree calculation for an area, when a vertex V corresponding to a Network-LSA is added to the shortest-path tree and its adjacent vertex W (joined by a link in V’s corresponding Network LSA), the cost from V to W, which is W’s network-to-router cost, is determined as follows:

- o For OSPFv2, if vertex W has a corresponding Extended-Link Opaque LSA with an Extended Link TLV for the link from W to V, and the Extended Link TLV has a Network-to-Router Metric Sub-TLV for the corresponding topology, then the cost from V to W is the metric in the Sub-TLV. Otherwise, the cost is 0.

- o For OSPFv3, if vertex W has a corresponding E-Router-LSA with a Router-Link TLV for the link from W to V, and the Router-Link TLV has a Network-to-Router Metric Sub-TLV, then the cost from V to W is the metric in the Sub-TLV. If not, the cost is 0.

3.5. Backward Compatibility

Due to the change of procedures in the SPF calculation, all routers in an area that includes one or more two-part metric networks must support the changes specified in this document. To ensure that, if an area is provisioned to support two-part metric networks, all routers supporting this capability must advertise a Router Information (RI) LSA with a Router Functional Capabilities TLV [acee-ospf-rfc4970bis] that includes the following Router Functional Capability Bit:

Bit	Capabilities
0	OSPF Two-part Metric [TPM]

Upon detecting the presence of a reachable Router-LSA without a companion RI LSA that has the bit set, all routers MUST disable the two-part metric functionalities and take the following actions:

- o If this router currently advertises network-to-router costs, remove the Network-to-Router Metric Sub-TLVs. This may lead to removal of parent TLVs and even withdrawal of the parent LSAs.

- o Recalculate routes w/o considering any network-to-router costs.

4. IANA Considerations

This document requests IANA to assigna a new bit in the Router Functional Capabilities TLV to indicate the capability of supporting two-part metric, a new Sub-TLV in the OSPF Extended-Link TLV Sub-TLV Registry, and a new Sub-TLV in the The OSPFv3 Extend-LSA Sub-TLV registry.

5. Security Considerations

This document does not introduce new security risks.

6. Acknowledgements

The authors would like to thank Abhay Roy, Hannes Gredler, Peter Psenak and Eric Wu for their comments and suggestions.

7. References

7.1. Normative References

[I-D.acee-ospf-rfc4970bis] Lindem, A., Shen, N., Vasseur, J., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", draft-acee-ospf-rfc4970bis-00 (work in progress), July 2014.

[I-D.ietf-ospf-lsa-extend] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPFv2 LSA Extendibility", draft-ietf-ospf-lsa-extend-00 (work in progress), August 2014.

[I-D.ietf-ospf-ospfv3-lsa-extend] Lindem, A., Mirtorabi, S., Roy, A., and F. Baker, "OSPFv3 LSA Extendibility", draft-ietf-ospf-ospfv3-lsa-extend-04 (work in progress), September 2014.

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, June 2007.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

[RFC5613] Zinin, A., Roy, A., Nguyen, L., Friedman, B., and D. Yeung, "OSPF Link-Local Signaling", RFC 5613, August 2009.

7.2. Informative References

[RFC6845] Sheth, N., Wang, L., and J. Zhang, "OSPF Hybrid Broadcast and Point-to-Multipoint Interface Type", RFC 6845, January 2013.

Authors' Addresses

Jeffrey Zhang
Juniper Networks, Inc.
10 Technology Park Drive
Westford, MA 01886

EMail: zzhang@juniper.net

Lili Wang
Juniper Networks, Inc.
10 Technology Park Drive
Westford, MA 01886

EMail: liliw@juniper.net

Acee Lindem
Cisco Systems
301 Midenhall Way
Cary, NC 27513

EMail: acee@cisco.com

David Dubois
General Dynamics C4S
400 John Quincy Adams Road
Taunton, MA 02780

EMail: dave.dubois@gdc4s.com

Vibhor Julka
L3 Communications, Linkabit
9890 Towne Centre Drive
San Diego, CA 92121

EMail: vibhor.julka@l-3Com.com

Tom McMillan
L3 Communications, Linkabit
9890 Towne Centre Drive
San Diego, CA 92121

EMail: tom.mcmillan@l-3com.com