Shaofu. Peng
Ran. Chen
ZTE Corporation
Ketan. Talaulikar
Arrcus, Inc
Peter. Psenak
Cisco Systems
26 May 2023

Algorithm Related IGP-Adjacency SID Advertisement
draft-ietf-lsr-algorithm-related-adjacency-sid-05

Abstract

Segment Routing architecture supports the use of multiple routing algorithms, i.e., different constraint-based shortest-path calculations can be supported. There are two standard algorithms: SPF and Strict-SPF, defined in Segment Routing architecture. There are also other user defined algorithms according to Flex-algo applicaiton. However, an algorithm identifier is often included as part of a Prefix-SID advertisement, that maybe not satisfy some scenarios where multiple algorithm share the same link resource. This document complement that the algorithm identifier can be also included as part of a Adjacency-SID advertisement.

Table of Contents

1. Introduction

Segment Routing architecture [RFC8402] supports the use of multiple routing algorithms, i.e., different constraint-based shortest-path calculations can be supported. There are two standard algorithms, i.e., SPF and Strict-SPF, that defined in Segment Routing architecture. For SPF, the packet is forwarded along the well known ECMP-aware Shortest Path First (SPF) algorithm employed by the IGPs. However, it is explicitly allowed for a midpoint to implement another forwarding based on local policy. For Strict Shortest Path First (Strict-SPF), it mandates that the packet be forwarded according to

the ECMP-aware SPF algorithm and instructs any router in the path to ignore any possible local policy overriding the SPF decision.

There are also other user defined algorithms according to IGP Flex Algorithm [RFC9350]. IGP Flex Algorithm proposes a solution that allows IGPs themselves to compute constraint based paths over the network, and it also specifies a way of using Segment Routing (SR) Prefix-SIDs and SRv6 locators to steer packets along the constraint-based paths. It specifies a set of extensions to ISIS, OSPFv2 and OSPFv3 that enable a router to send TLVs that identify (a) calculation-type, (b) specify a metric-type, and (c) describe a set of constraints on the topology, that are to be used to compute the best paths along the constrained topology. A given combination of calculation-type, metric-type, and constraints is known as an FAD (Flexible Algorithm Definition).

However, an algorithm identifier is often included as part of a Prefix-SID advertisement, that maybe not satisfy some scenarios where multiple algorithm share the same link resource. In addition to Prefix-SID, this document complement that the algorithm identifier can be also included as part of an Adjacency-SID advertisement for SR-MPLS, so that each Flex-algo plane corresponding to different algorithm types can be allocated with a dedicated segment ID related to the corresponding algorithm type.

2. Requirements Language

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

3. Use-cases

The algorithm related SID can be used to apply different treatments for packets classified into different algorithms. Some known treatments are described in the following use cases:

* One use-case is that a TI-LFA backup path computed in Flex-algo plane may contain Adjacency Segments and require to contain an algorithm-aware Adjacency-SID, which can not only steer the traffic towards the link, but also distinguish traffic between different algorithms. Benefit from this, for the protected Adjacency-SID which belongs to a TI-LFA path within specific Flex-algo plane, the backup path of such Adjacency-SID can continue to follow the algorithm specific constraints that is consistent with the primary path.

* Another use-case is to help the enhancement PHB (per hop behavior) related with specific algorithm type on the data plane. Generally, QoS policies related to the algorithm type of each Flex-algo plane can be configured and installed, and the packets can be forwarded based one the algorithm related SID and QoS policy.

* Another use-case is to help the enhancement OAM related with specific algorithm type on the control plane and data plane, such as statistics of traffic of different algorithms on the same link, ping/traceroute detection (or other tools) for specific algorithm.

There may be other potential use cases. Note that the specification details of these treatments are beyond the scope of this document. This document only provides foundation required for these treatments.

4. Adjacency Segment Identifier per Algorithm

This section describe that the algorithm related segment ID is flooded through the IGP protocol.

4.1. ISIS Adjacency Segment Identifier per Algorithm

[RFC8667] describes the IS-IS extensions that need to be introduced for Segment Routing operating on an MPLS data plane. It defined Adjacency Segment Identifier (Adj-SID) sub-TLV advertised with TLV-22/222/23/223/141, and Adjacency Segment Identifier (LAN-Adj-SID) Sub-TLV advetised with TLV-22/222/23/223. Accordingly, this document defines two new optional Sub-TLVs, "ISIS Adjacency-SID per Algorithm Sub-TLV" and "ISIS LAN Adjacency-SID per Algorithm Sub-TLV", which contains a field representing the algorithm type.

4.1.1. ISIS Adjacency-SID per Algorithm Sub-TLV

ISIS Adjacency-SID per Algorithm 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     |     Weight    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Algorithm   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         SID/Label/Index (variable)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: ISIS Adjacency-SID per Algorithm Format

where:

Type: TBA1.

Length: 6 or 7 depending on size of the SID.

Flags: Refer to Adjacency Segment Identifier (Adj-SID) sub-TLV.

Weight: Refer to Adjacency Segment Identifier (Adj-SID) sub-TLV.

Algorithm: The Algorithm field contains the identifier of the algorithm the router uses to apply algorithm specific treatment configured on the adjacency.

SID/Label/Index: Refer to Adjacency Segment Identifier (Adj-SID) sub-TLV.

For a P2P link, an SR-capable router MAY allocate different Adjacency-SIDs for different algorithms, if this link participates in the plane related to different algorithms.

4.1.2. ISIS LAN Adjacency-SID per Algorithm Sub-TLV

ISIS LAN Adjacency-SID per Algorithm 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     |    Weight     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Algorithm   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Neighbor System-ID (ID length octets)         |
+                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SID/Label/Index (variable)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: ISIS LAN Adjacency-SID per Algorithm Format

where:

Type: TBA2.

Length: Variable.

Flags: Refer to Adjacency Segment Identifier (LAN-Adj-SID) Sub-TLV.

Weight: Refer to Adjacency Segment Identifier (LAN-Adj-SID) Sub-TLV.

Algorithm: The Algorithm field contains the identifier of the algorithm the router uses to apply algorithm specific treatment configured on the adjacency.

SID/Label/Index: Refer to Adjacency Segment Identifier (LAN-Adj-SID) Sub-TLV.

For a broadcast link, an SR-capable router MAY allocate different Adjacency-SIDs for different algorithms, if this link participates in the plane related to different algorithms.

4.2. OSPFv2 Adjacency Segment Identifier per Algorithm

[RFC8665] describes the OSPFv2 extensions that need to be introduced for Segment Routing operating on an MPLS data plane. It defined Adj-SID Sub-TLV and LAN Adj-SID Sub-TLV advertised with Extended Link TLV defined in [RFC7684]. Accordingly, this document defines two new optional Sub-TLVs, "OSPFv2 Adjacency-SID per Algorithm Sub-TLV" and "OSPFv2 LAN Adjacency-SID per Algorithm Sub-TLV", which contains a field representing the algorithm type.

4.2.1. OSPFv2 Adjacency-SID per Algorithm Sub-TLV

OSPFv2 Adjacency-SID per Algorithm 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     |   Algorithm   |     MT-ID     |  Weight       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   SID/Label/Index (variable)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: OSPFv2 Adjacency-SID per Algorithm Format

where:

Type: TBA3

Length: 7 or 8 octets, depending on the V-Flag.

Flags: Refer to OSPFv2 Adj-SID Sub-TLV.

Algorithm: The Algorithm field contains the identifier of the algorithm the router uses to apply algorithm specific treatment configured on the adjacency.

MT-ID: Refer to OSPFv2 Adj-SID Sub-TLV.

Weight: Refer to OSPFv2 Adj-SID Sub-TLV.

SID/Index/Label: Refer to OSPFv2 Adj-SID Sub-TLV.

For a P2P link, an SR-capable router MAY allocate different Adjacency-SIDs for different algorithms, if this link participates in the plane related to different algorithms.

4.2.2. OSPFv2 LAN Adjacency-SID per Algorithm Sub-TLV

OSPFv2 LAN Adjacency-SID per Algorithm 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     |   Algorithm   |     MT-ID     |    Weight     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Neighbor ID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    SID/Label/Index (variable)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: OSPFv2 LAN Adjacency-SID per Algorithm Format

where:

Type: TBA4

Length: 11 or 12 octets, depending on the V-Flag.

Flags: Refer to OSPFv2 LAN Adjacency-SID Sub-TLV.

Algorithm: The Algorithm field contains the identifier of the algorithm the router uses to apply algorithm specific treatment configured on the adjacency.

MT-ID: Refer to OSPFv2 LAN Adj-SID Sub-TLV.

Weight: Refer to OSPFv2 LAN Adj-SID Sub-TLV.

Neighbor ID: Refer to OSPFv2 LAN Adj-SID Sub-TLV.

SID/Index/Label: Refer to OSPFv2 LAN Adj-SID Sub-TLV.

For a broadcast link, an SR-capable router MAY allocate different Adjacency-SIDs for different algorithms, if this link participates in the plane related to different algorithms.

4.3. OSPFv3 Adjacency Segment Identifier per Algorithm

[RFC8666] describes the OSPFv3 extensions that need to be introduced for Segment Routing operating on an MPLS data plane. It defined Adj-SID Sub-TLV and LAN Adj-SID Sub-TLV advertised with Router-Link TLV as defined in [RFC8362]. Accordingly, this document defines two new optional Sub-TLVs, "OSPFv3 Adjacency-SID per Algorithm Sub-TLV" and "OSPFv3 LAN Adjacency-SID per Algorithm Sub-TLV", which contains a field representing the algorithm type.

4.3.1. OSPFv3 Adjacency-SID per Algorithm Sub-TLV

OSPFv3 Adjacency-SID per Algorithm 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     |     Weight    |   Algorithm   |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SID/Label/Index (variable)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: OSPFv3 Adjacency-SID per Algorithm Format

where:

Type: TBA5

Length: 7 or 8 octets, depending on the V-Flag.

Flags: Refer to OSPFv3 Adj-SID Sub-TLV.

Weight: Refer to OSPFv3 Adj-SID Sub-TLV.

Algorithm: The Algorithm field contains the identifier of the algorithm the router uses to apply algorithm specific treatment configured on the adjacency.

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

SID/Index/Label: Refer to OSPFv3 Adj-SID Sub-TLV.

For a P2P link, an SR-capable router MAY allocate different Adjacency-SIDs for different algorithms, if this link participates in the plane related to different algorithms.

4.3.2. OSPFv3 LAN Adjacency-SID per Algorithm Sub-TLV

OSPFv3 LAN Adjacency-SID per Algorithm 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     |     Weight    |   Algorithm   |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Neighbor ID                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    SID/Label/Index (variable)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: OSPFv3 LAN Adjacency-SID per Algorithm Format

where:

Type: TBA6

Length: 11 or 12 octets, depending on the V-Flag.

Flags: Refer to OSPFv3 LAN Adj-SID Sub-TLV.

Weight: Refer to OSPFv3 LAN Adj-SID Sub-TLV.

Algorithm: The Algorithm field contains the identifier of the algorithm the router uses to apply algorithm specific treatment configured on the adjacency.

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

Neighbor ID: Refer to OSPFv3 LAN Adj-SID Sub-TLV.

SID/Index/Label: Refer to OSPFv3 LAN Adj-SID Sub-TLV.

For a broadcast link, an SR-capable router MAY allocate different Adjacency-SIDs for different algorithms, if this link participates in the plane related to different algorithms.

5. Procedures

The method introduced in this document enables the traffic of different flex-algo plane to be distinguished when they are routed on the same link, and can be applied with different local treatment (such as providing different repair paths, traffic statistics, QoS policies, etc) per algorithm.

A node may, according to each flex-algo plane (corresponding to the specific algorithm types) which it participated in, allocate segment identifiers corresponding to each algorithm types, and these segment identifiers need to be flooded through IGP. As defined in Section 4, algorithm field must be encoded in the flooded packet.

Depending on implementation, SIDs allocation is generally triggered by configuration. For algorithm specific Adjacency-SID, one of the difficulties is that during this configuration phase it is not straightforward for a link to be included in an Flex-algo plane, as this can only be determined after all nodes in the network have negotiated the FAD. Note that Node-SID per algorithm may also face similar difficulties (considering the abnormal situation where nodes have to stop participating in the flex-algo plane after FAD negotiation, referring to section 5.3 of [RFC9350]).

Developers can flexibly refer to any of the following implementation choices.

* One choice is that as long as an IGP instance with an algorithm enabled for a level/area is configured on the node, the node may allocate Adjacency-SIDs for that algorithm statically for all links joined to that level/area. Similar way may be also applied to node-SID per algorithm. That is, algorithm specific SID can be allocated regardless of the felx-algo participation and wining

FAD. If the router stops participating, or the link is excluded from the flex-algo, the advertised algorithm specific SID does not cause any issue, but is just not used.

* Another choice is to allocate and withdraw algorithm specific Adjacency-SID dynamically according to the result of FAD negotiation, i.e., algorithm specific Adjacency-SID is allocated and advertised only for those links that have joined the Flex-algo plane. Similar choice may be also applied to node-SID per algorithm. This choice is RECOMMENDED.

The RECOMMENDED implementation choice also make sense for other type of states per algorithm. A node may, according to each flex-algo plane (corresponding to the specific algorithm types) which it participated in, config local treatments (such as repair paths, traffic statistics counters, QoS policies, etc) corresponding to each algorithm types and apply them to the links that participated in the corresponding flex-algo plane. In this case, the node may dynamically create or delete these local treatments according to the result of FAD negotiation.

The (LAN) Adjacency-SID per Algorithm Sub-TLV MUST not be used to advertise algorithm 0 specific SIDS.

Note that the advertisement specification defined in Section 4 does not have any requirements for the SID allocation rules. Some particular advertisement method based on particular allocation rules are not within the scope of this document.

Once the node originates an algorithm specific Adjacency-SID and sends it to the network, the coresponding local SID entry (i.e., an MPLS label forwarding entry) must be installed on the fowarding plane. The local SID entry, combined with local treatments (such as QoS polices), are used to continue to forward data packets in the context of the specific algorithm.

A node may receive different algo-SIDs (corresponding to different algorithm types with the related flex-algo plane) originated from other nodes and flooded by IGP. As defined in Section 4, the algorithm field can be gotten from the flooded packet to indicate algorithm specific SIDs. Then, algo-SIDs, with other SIDs, are maintained in the link state database.

If the received algorithm is not within the range [128,255], the related (LAN) Adjacency-SID per Algorithm Sub-TLV MUST be ingored.

When a node receives a forwarding data packet whose active segment is an algorithm specific Adjacency-SID and matches the coresponding local SID entry, the node forwards the data packet to the corresponding outgoing port and applies algorithm related local treatments (such as QoS policies) to the packet. The local treatments may also be applied for the case of algorithm specific Node-SID.

5.1. Examples of Algorithm Specific Adjacency-SID

The following figure shows an example of algorithm specific Adjacency-SID.

```
        [S1]--------[D]--------[S2]
         |           |          |
         |           |          |
         |           |          |
        [A]---------[B]--------[C]
```

Figure 7: Flex-algo LFA Path with Algorithm Specific Adjacency-SID

Suppose that node S1, A, B, D and their inter-connected links belongs to FA-id 128 plane, and S2, B, C, D and their inter-connected links belongs to FA-id 129 plane. The IGP metric of link B-D is 100, and all other links have IGP metric 1. Both FA-id 128 and 129 use IGP default metric type for path calculation. In FA-id 128 plane, from S1 to destination D, the primary path is S1-D, and the TI-LFA backup path is segment list {node(B), adjacency(B-D)}. Similarly, In FA-id 129 plane, from S2 to destination D, the primary path is S2-D, and the TI-LFA backup path is segment list {node(B), adjacency(B-D)}. The above TI-LFA path of FA-id 128 plane can be translated to {node-SID(B)@FA-id128, adjacency-SID(B-D)@FA-id128}, and TI-LFA path of FA-id 129 plane will be translated to {node-SID(B)@FA-id129, adjacency-SID(B-D)@FA-id129}. So that node B can distinguish the flows of FA-id 128 and FA-id 129 based on different adjacency-SID(B-D) and their related label forwarding entries, and take different treatments of them when they are forwarded to the same outgoing link B-D.

6. Deployment Considerations

When multiple flex-algos are deployed in the network and they share the same link, multiple algorithm specific Adjacency-SIDs may need to be allocated on such a link, to distinguish the traffic of different algorithms and provide possible different treatment.

Even if a link is only used by a single flex-algo, because the link always belongs to algorithm 0 by default, both the traditional Adjacency-SID (termd as adj-sid@algo-0) and the algorithm specific Adjacency-SID (termd as adj-sid@algo-x) may need to be allocated on that link, so that the potentional repair paths of the two Adjacency-SIDs can be distinguished.

If the topology of multiple flex-algo planes, and physical topology, are isomorphic, that is, they contain the same nodes and same inter-connected links, but due to the differences between these FADs (such as different metric types), different repair paths will also be calculated on the same topology. Therefore, multiple algorithm specific Adjacency-SIDs may still need to be provided on the same link.

It is not recommended to bind a link to algorithm 1 (Strict SPF) and allocate adj-sid@algo-1. Such Adjacency-SID is no useful.

The operator may configure the policy on the node to turn off the algorithm specific processing capability for each algorithm, and the node will not allocate algorithm specific Adjacency-SIDs on the links those joined to the flex-algo plane, this is a local behavior. As mentioned before, the algorithm specific processing capability can be further subdivided into repair path per algorithm, statistics per algorithm, QoS policy per algorithm, etc. Assuming that a node wants to support the capability of repair path per algorithm, in this case, for an individual link, it is also controlled by the adjacency backup capability. When adjacency backup is disabled, it will let the capablitiy of repair path per algorithm be also invalid, so the link does not need to allocate algorithm specific Adjacency-SIDs.

In any case, when instantiate a segment list (such as a TI-LFA path) within a specific flex-algo plane, for each Adjacency Segment of that list, if it has a corresponding algorithm specific Adjacency-SID, the algorithm specific Adjacency-SID MUST be used to construct SID list; if it has not, traditional Adjacency-SID can be used.

7. IANA Considerations

7.1. IANA ISIS Considerations

This document makes the following registrations in the "Sub-TLVs for TLV 22, 23, 25, 141, 222, and 223" registry.

Type	Description	22	23	25	141	222	223
TBA1	Adjacency-SID per Algorithm	y	y	n	y	y	y
TBA2	LAN Adjacency-SID per Algorithm	y	y	n	y	y	y

7.2. IANA OSPFv2 Considerations

This document makes the following registrations in the OSPFv2 Extended Link TLV Sub-TLVs Registry.

Value	Description	Reference
TBA3	OSPFv2 Adjacency-SID per Algorithm Sub-TLV	This document
TBA4	OSPFv2 LAN Adjacency-SID per Algorithm Sub-TLV	This document

7.3. IANA OSPFv3 Considerations

This document makes the following registrations in the "OSPFv3 Extended-LSA Sub-TLVs" Registry.

Value	Description	Reference
TBA5	OSPFv3 Adjacency-SID per Algorithm Sub-TLV	This document
TBA6	OSPFv3 LAN Adjacency-SID per Algorithm Sub-TLV	This document

8. Security Considerations

There are no new security issues introduced by the extensions in this document. Refer to [RFC8665], [RFC8666], [RFC8667] for other security considerations.

9. Acknowledgements

We would like to thank Aijun Wang, Robert Raszuk, Gyan Mishra, Jie Dong and Xuesong Geng for their reviews and discussions to the content of this document.

10. Contributors

The following people gave a substantial contribution to the content of this document.

Les Ginsberg
Cisco Systems, Inc.
United States of America
Email: ginsberg@cisco.com

11. Normative References

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

[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>.

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

[RFC8362] Lindem, A., Roy, A., Goethals, D., Reddy Vallem, V., and F. Baker, "OSPFv3 Link State Advertisement (LSA) Extensibility", RFC 8362, DOI 10.17487/RFC8362, April 2018, <https://www.rfc-editor.org/info/rfc8362>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

[RFC8665] Psenak, P., Ed., Previdi, S., Ed., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", RFC 8665, DOI 10.17487/RFC8665, December 2019, <https://www.rfc-editor.org/info/rfc8665>.

[RFC8666] Psenak, P., Ed. and S. Previdi, Ed., "OSPFv3 Extensions for Segment Routing", RFC 8666, DOI 10.17487/RFC8666, December 2019, <https://www.rfc-editor.org/info/rfc8666>.

[RFC8667] Previdi, S., Ed., Ginsberg, L., Ed., Filsfils, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", RFC 8667, DOI 10.17487/RFC8667, December 2019, <https://www.rfc-editor.org/info/rfc8667>.

[RFC9350] Psenak, P., Ed., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", RFC 9350, DOI 10.17487/RFC9350, February 2023, <https://www.rfc-editor.org/info/rfc9350>.

Authors' Addresses

Shaofu Peng
ZTE Corporation
China
Email: peng.shaofu@zte.com.cn

Ran Chen
ZTE Corporation
China
Email: chen.ran@zte.com.cn

Ketan Talaulikar
Arrcus, Inc
India
Email: ketant.ietf@gmail.com

Peter Psenak
Cisco Systems
Slovakia
Email: ppsenak@cisco.com