SEGMENT ROUTING

Part I

Clarence Filsfils
Kris Michielsen
Ketan Talaulikar

COPYRIGHT

Segment Routing, Part I
by Clarence Filsfils, Kris Michielsen, Ketan Talaulikar

Kindle Edition v1.2, October 2016

Warning and Disclaimer

Trademark Acknowledgments

Feedback Information

The authors' goal is to create an in-depth technical book of the highest quality and value. It is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers' feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us via the blog http://segment-routing-book.blogspot.com or via email at segment-routing-book@googlegroups.com. Please make sure to include the book title in your message.

We greatly appreciate your assistance.

Segment Routing, Part I

TABLE OF CONTENTS

COPYRIGHT

CONVENTIONS USED IN THIS BOOK

1 INTRODUCTION

1.1 Objectives of the Book

1.2 Why Did We Start SR?

1.3 The SDN and OpenFlow Influences

1.4 100% Coverage for IPFRR and Optimum Repair Path

1.5 Other Benefits

1.6 Team

1.7 Keeping Things Simple

1.8 Standardization and Multi-Vendor Consensus

1.9 Global Label

1.10 SR MPLS

1.11 SRv6

1.12 Industry Benefits

1.13 References

2 FUNDAMENTALS OF SEGMENT ROUTING

2.1 What is Segment Routing?

2.2 Concept of Segments

2.2.1 Segments and Segment Identifiers

2.2.2 Combining Segments

2.2.3 Segment List Operations

2.2.4 Global and Local Segments

2.2.4.1 Global Segment
2.2.4.2 Local Segment
2.3 Segment Routing Control Plane
2.3.1 IGP Segments
2.3.1.1 IGP Prefix Segment
2.3.1.1.1 IGP Node Segment
2.3.1.1.2 IGP Anycast Segment
2.3.1.2 IGP Adjacency Segment
2.3.1.2.1 Layer-2 Adjacency-SID
2.3.1.2.2 Group Adjacency-SID
2.3.2 BGP Segments
2.3.2.1 BGP Prefix Segment
2.3.2.1.1 BGP Anycast Segment
2.3.2.2 BGP Peer Segments
2.3.2.2.1 BGP Peer Node Segment
2.3.2.2.2 BGP Peer Adjacency Segment
2.3.2.2.3 BGP Peer Set Segment
2.3.3 Combining Segments
2.4 Segment Routing Data Plane
2.5 Summary
2.6 References
3 SEGMENT ROUTING MPLS DATA PLANE
3.1 IPv4 and IPv6
3.2 Existing MPLS Data Plane
3.3 Segment Routing Global Block
3.4 SR MPLS Label Stack Operations
3.4.1 MPLS Label Push or Imposition
3.4.2 MPLS Label Swap
3.4.3 Penultimate Hop Node Behavior

3.4.3.1 Swap with Prefix-SID Label
3.4.3.2 Pop Prefix-SID Label
3.4.3.3 Swap with Explicit-Null Label
3.4.4 Final Hop Node
3.4.5 Adjacency-SIDs and MPLS
3.4.6 Forwarding Entry with Unlabelled
3.4.7 IGP SR MPLS Forwarding Entries
3.5 MPLS TTL and TC (or EXP) Treatment
3.5.1 TTL Processing IP-to-MPLS and MPLS-to-IP
3.5.2 TTL Processing MPLS-to-MPLS
3.5.3 Handling MPLS TTL Expiry
3.5.4 TC or EXP Field Processing
3.6 MPLS Load-Balancing
3.7 MPLS MTU Considerations
3.8 Summary
3.9 References
4 MANAGEMENT OF THE SRGB
4.1 SRGB Size
4.2 Multi-Range SRGB
4.3 SRGB Label Range
4.4 SRGB and Anycast Segments
4.5 SRGB Configuration
4.5.1 Cisco IOS XR SRGB Implementation
4.5.2 SRGB Errors
4.6 Summary
4.7 References
5 SEGMENT ROUTING IGP CONTROL PLANE
5.1 Segment Routing Extensions for IGPs
5.1.1 Segment Routing for IS-IS

5.1.1.1 IS-IS Protocol Extensions
5.1.1.2 ISIS SR Configuration
5.1.2 Segment Routing for OSPFv2
5.1.2.1 OSPFv2 Protocol Extensions
5.1.2.2 OSPFv2 SR Configuration
5.1.3 Segment Routing for OSPFv3
5.2 Segment Routing Capabilities
5.2.1 SRGB Advertisement
5.2.2 SR Capabilities Advertisement in IS-IS
5.2.3 SR Capabilities Advertisement in OSPFv2
5.3 Prefix-SID
5.3.1 Signaling Prefix-SID in IGP Advertisements
5.3.2 Prefix-SID Configuration
5.3.2.1 Prefix-SID with N-flag Unset
5.3.2.2 Prefix-SID Penultimate Hop Behavior
5.3.3 Prefix-SID Advertisement in IS-IS
5.3.3.1 ISIS Prefix-SID with N-flag Unset
5.3.3.2 ISIS Prefix-SID with Explicit-null Flag
5.3.4 Prefix-SID Advertisement in OSPFv2
5.3.4.1 OSPF Prefix-SID with N-flag Unset
5.3.4.2 OSPF Prefix-SID with Explicit-null Flag
5.4 Adjacency-SID
5.4.1 Signaling Adjacency-SID in IGP Advertisements
5.4.2 Adjacency-SID Allocation
5.4.3 LAN Adjacency-SID
5.4.3.1 IGP LAN Adjacencies Reminder
5.4.3.2 LAN Adjacency Segments
5.4.4 Adjacency-SID Persistency
5.4.5 ISIS Adjacency-SID

5.4.5.1 IS-IS Adjacency-SID Advertisement
5.4.5.2 IS-IS LAN Adjacency-SID
5.4.5.3 ISIS LAN Adjacency-SID Advertisement
5.4.6 OSPFv2 Adjacency-SID
5.4.6.1 OSPFv2 Adjacency-SID Advertisement
5.4.6.2 OSPF Multi-AreaAdjacency-SID
5.4.6.3 OSPFv2 LAN Adjacency-SID
5.4.6.4 OSPF LAN Adjacency-SID Advertisement
5.5 IGP Multi-Area/Level Operations
5.5.1 Determining Penultimate Hop for a Prefix
5.5.1.1 ISIS Determination of Penultimate Hop with Multi-Level
5.5.1.2 OSPF Determination of Penultimate Hop with Multi-Area
5.5.2 Propagation of Prefix-SID Across Areas/Levels
5.5.3 IS-IS Multi-Area Example
5.5.4 OSPFv2 Multi-Area example
5.6 Redistribution with Prefix-SIDs
5.6.1 Redistribution Examples
5.6.1.1 Redistributing between IGPs
5.6.1.2 Redistributing between IGP and BGP
5.6.2 OSPF NSSA
5.7 Summary
5.8 References
6 SEGMENT ROUTING BGP CONTROL PLANE
6.1 BGP Labeled Unicast
6.2 BGP Prefix-SID
6.3 BGP Prefix-SID Advertisement
6.4 Interoperability with non-SR BGP-LU

6.5 SR BGP in Seamless MPLS Architecture
6.6 BGP based Data Center Reminder
6.6.1 From Classic DC to Clos Topology
6.6.2 BGP in Layer 3 Data Center
6.6.3 MPLS in Data Center
6.6.4 BGP Labeled Unicast in Data Center
6.7 SR Benefits in DC
6.7.1 Load-Balancing Efficiency
6.7.2 Non-Oblivious Routing
6.7.3 Performance-Aware Routing
6.7.4 Deterministic Network Probing
6.8 BGP Prefix-SID in Data Center
6.8.1 Configuring E-BGP Prefix Segments
6.8.2 Configuring L3VPN Overlay Service
6.8.3 Configuring ECMP with E-BGP Multipath
6.8.4 Configuring I-BGP Prefix Segments
6.9 Summary
6.10 References
7 SEGMENT ROUTING IN EXISTING MPLS NETWORKS
7.1 Co-Existence of SR and other MPLS Protocols
7.1.1 Control Plane Co-Existence
7.1.2 Data Plane Co-Existence
7.1.2.1 MPLS-to-MPLS, MPLS-to-IP
7.1.2.2 IP-to-MPLS
7.1.3 Implementation Aspects
7.2 Segment Routing and LDP Interworking
7.2.1 LDP to SR Interworking
7.2.1.1 LDP to SR Implementation Aspects
7.2.1.2 LDP to SR Illustration

7.2.2 SR to LDP Interworking
7.2.2.1 SR to LDP Implementation Aspects
7.2.2.2 SR to LDP Illustration
7.2.3 SR over LDP, LDP over SR
7.2.4 SR/LDP Interworking Summary
7.3 Summary
7.4 References
8 SEGMENT ROUTING MAPPING SERVER
8.1 Function of Mapping Server
8.2 Position of Mapping Server in Network
8.3 Function of Mapping Client
8.4 Mapping Server Architecture
8.5 Mapping Server Local Policy Configuration
8.6 Mapping Ranges Conflict Resolution
8.6.1 Overlapping and Conflicting Mapping Ranges Examples
8.6.2 Selecting Mapping Ranges from Different Mapping Servers
8.6.3 Mapping Range Conflict with "Native" Prefix-SID
8.7 IGP Configuration for SRMS
8.7.1 Mapping Server
8.7.2 Mapping Client
8.7.3 Mapping Server and Mapping Client
8.8 Mapping Server Advertisements in IS-IS
8.9 Mapping Server Advertisements inOSPF
8.10 Mapping Server in Multi-Area/LevelNetwork
8.10.1 ISIS Inter-Level SRMS Advertisements
8.10.2 OSPF Inter-Area SRMS Advertisements
8.11 Summary
8.12 References

9 TOPOLOGY INDEPENDENT LFA (TI-LFA)
9.1 Introduction
9.2 Loop-Free Alternate
9.3 Remote LFA
9.4 Topology Independent LFA
9.4.1 TI-LFA Computation
9.4.2 Segment List Size Analysis
9.4.3 TE Infrastructure for Repair Paths
9.5 TI-LFA Protection Options
9.5.1 Link Protection
9.5.1.1 ISIS Configuration
9.5.1.2 OSPF Configuration
9.5.1.3 Link Protection Examples
9.5.2 Node and SRLG Protection
9.5.2.1 ISIS Configuration
9.5.2.2 OSPF Configuration
9.5.2.3 Node Protection Example
9.5.2.4 SRLG Protection Example
9.5.3 De Facto Node Protection
9.6 Addressing P and PQ Nodes
9.6.1 Addressing P and PQ Nodes in ISIS
9.6.2 Addressing P and PQ Nodes in OSPF
9.7 Protecting Adjacency Segments
9.8 TI-LFA for IP and LDP Traffic
9.8.1 Protecting Unlabeled IP Traffic
9.8.2 Protecting LDP Traffic
9.8.3 TI-LFA and SR/LDP Interworking
9.9 TI-LFA Load-Balancing on RepairPath
9.10 Microloop Avoidance

9.10.1 An Overview of Microloops
9.10.2 Existing Microloops Avoidance Mechanisms
9.10.3 Microloop Avoidance with Segment Routing
9.11 Summary
9.12 References
10 LARGE SCALE INTERCONNECT WITH SEGMENT ROUTING
10.1 Applicability Considerations
10.2 Reference Design
10.2.1 Interconnecting Nodes
10.2.2 Interconnecting Endpoints
10.3 Design Options
10.3.1 Leaf and Core Domains Sizing
10.3.2 Sub-Leaf Domains
10.3.3 Traffic Engineering
10.4 Scale Example
10.5 Deployment Models
10.6 Benefits
10.7 Summary
10.8 References
11 VERIFYING CONNECTIVITY IN SR MPLS NETWORK
11.1 Existing IP Toolkit
11.1.1 IP Ping
11.1.2 IP Traceroute
11.1.3 IP Ping, Traceroute and ECMP
11.1.4 IP Ping in MPLS Environment
11.1.5 IP Traceroute in MPLS Environment
11.2 Existing MPLS Toolkit
11.2.1 MPLS ping
11.2.2 MPLS Echo Request/Reply Packet

11.2.3 MPLS Traceroute
11.2.4 Path Discovery using MPLS Traceroute
11.2.5 MPLS Ping and Traceroute using Nil-FEC
11.3 MPLS OAM for Segment Routing
11.4 Summary
11.5 References
12 SEGMENT ROUTING IPV6 DATE PLANE
12.1 “Segment” in IPv6
12.2 Segment Identifiers in SRv6
12.3 IPv6 Headers Reminder
12.4 Routing Header
12.5 Segment Routing Header
12.6 The SRH Procedures
12.6.1 Source Node
12.6.2 Segment Endpoint Node
12.6.3 Transit Node
12.7 SRH Insertion versus Pushing IPv6 Header
12.7.1 SRH Insertion at Source
12.7.2 SRH Insertion at Ingress
12.7.3 Encap Header Insertion at Ingress
12.8 Summary
12.9 References
SUMMARY

CONVENTIONS USED IN THIS BOOK

Textual Conventions

The books has multiple flows:

- General flow: This is the regular flow of content that a reader wanting to learn SR would follow. It contains facts, no opinions and is objective in nature.
- Highlights: Highlight boxes emphasize important elements and topics for the reader. These are presented in a "highlight" box.

HIGHLIGHT

This is an example of a highlight.

- Opinions: This content expresses opinions, choices and tradeoffs. This content is not necessary to understand SR, but gives some more background to the interested reader and is presented as quotes. We have also invited colleagues in the industry who have been very active on the SR project to share their opinions on SR in general or some specific aspects. These quotes will also indicate the name of the persons providing that opinion.

> "This is an example opinion."
>
> *— John Doe*

- Reminders: The reminders briefly explain technological aspects (mostly outside of SR) that may not yet be known or forgotten by the reader. They are presented in a “reminder” box.

REMINDER

This is an example reminder box

Illustrations and Examples

The illustrations and examples in this book follow the following conventions:

- Router-id of NodeX is 1.1.1.X. Other loopbacks have address n.1.1.X, with n an index. For IPv6 that is 2001::1:1:1:X/128 and 2001::n:1:1:X/128, with n an index.
- Interface IPv4 address of an interface on NodeX connected to NodeY is 99.X.Y.X/24, with X<Y. E.g. a link connecting Node2 to Node3 has a network address 99.2.3.0/24; the interface address on Node2 is 99.2.3.2 and on Node3 it is 99.2.3.3.
- Interface IPv6 address of an interface on NodeX connected to NodeY is 2001::99:X:Y:X/112, with X<Y.
- Prefix-SIDs are labels in the range 16000 to 23999. This is the default Segment Routing Global Block in Cisco devices and common practice.
- Adjacency-SIDs are labels in the [30000-39999] range and have the format 3xxyy for an adjacency on xx going to yy. If there are multiple links between two nodes then we will use 3nxxyy, with n an index of each adjacency-SIDs. Visually 3nxxyy looks like 3xxyy.
- Other dynamic labels used for SR, such as Peering-SIDs, Binding-SIDs, …) are labels in the range [40,000-49,999].
- Dynamic labels allocated by other MPLS applications such as LDP, RSVP-TE, BGP3107 and so on, are in the range [90,000-99,999].

1 INTRODUCTION

1.1 Objectives of the Book

"Segment Routing – Part I" has several objectives:

- To teach the basic elements of Segment Routing (SR) with a focus on the IGP extensions, the IGP/LDP interaction, Topology-Independent Fast Reroute (TI-LFA FRR) and the MPLS data plane. "Segment Routing – Part II" will focus on the Traffic Engineering applications, both distributed and centralized, and the application of the SR architecture to the IPv6 data plane. Part I describes the SR use-cases that have been first considered for deployment. Part I lays the foundation for Part II.
- To provide design guidelines and illustrations based on real use-cases we have deployed
- To invite operators who have participated in this project to provide their view and offer tips based on their experience defining and deploying SR
- To explain why we have defined SR the way it is, what were our primary objectives, what were our intuitions, what happened during the first 3 years of this technology

The first objective is natural and we will dedicate lots of content to teach SR, let's say in an "objective" manner.

We believe that it is important to complement the "objective" understanding of a technology with a more "subjective" viewpoint. This helps to understand what part of the technology is really important in real life, what to optimize for, what trade-offs to make or not make.

The other three objectives are related to getting this subjective experience. By reviewing deployed use-cases, by incorporating the highlights of the operators, by explaining how we came up with SR and what were our initial intuitions and goals; we hope that the readers will get a much more practical understanding of the SR architecture.

The main part of the text is dedicated to the first goal, the objective explanation of the technology and the description of use-cases. Important concepts are reminded in textboxes titled “Highlight”.

To clearly distinguish the subjective content from the main flow of the book (objective), the subjective viewpoints will be inserted in text boxes attributed to the person providing the subjective opinion.

This entire first chapter should be considered as a subjective entry. It is written by Clarence to describe how he got started with SR, then the influence of SDN and OpenFlow, how he managed the project and got the close involvement of operators to define the technology and drive its implementation for specific use-cases.

We stress the subjective nature of this chapter and the opinion text boxes along this book. They express personal viewpoints to help the reader forge his own viewpoint. They are not meant to be “it must be this way” or “this is the only right way” type of guidelines. They are meant to clearly describe how some people think of the technology so that the readers can leverage their viewpoints to build their own. This is very different from the normal flow of the book where we try to be objective and describe the technology as it is (“then it is really like this”).

1.2 Why Did We Start SR?

In 1998, I was hired in the European consulting team at Cisco to deploy a new technology called tag-switching (later renamed MPLS).

This was a fantastic experience: witnessing the entire technology definition process, working closely with the MPLS architecture team, having first-hand experience designing, deploying the first and biggest MPLS networks and collecting feedback and requirements from operators.

Over these years, while the elegance of the MPLS data plane has rarely been challenged, it became obvious that the MPLS "classic" (LDP and RSVP-TE) control-plane was too complex and lacked scalability.

In 2016, when we write this text, it should be safe to write that LDP is redundant to the IGP and that it is better to distribute labels bound to IGP signaled prefixes in the IGP itself rather than using an independent protocol (LDP) to do it.

LDP adds complexity: it requires one more process to configure and manage and it creates complicated interaction problems with the IGP (LDP-IGP synchronization issue, RFC 5443, RFC 6138).

Clearly, LDP was invented 20 years ago for various reasons that were good at that time. We are not saying that mistakes were made in the 1990's. We are saying that, in our opinion[1], if an operator were to deploy a greenfield MPLS network in 2016, considering the issues described above and the experience learned with RLFA (RFC 7490), they would not think of using LDP and would prefer to distribute the labels directly in the

IGP. This requires straightforward IGP extensions as we will see in chapter 5, “Segment Routing IGP Control Plane”.

On the RSVP-TE side, from a bandwidth admission control viewpoint, it seems safe to write that there has been weak deployment and that the few who deployed have reported complex operation models and scaling issues. In fact, most of the RSVP-TE deployments have been limited to fast re-route (FRR) use-case.

Overall, we would estimate that 10% of the SP market and likely 0% of the Enterprise market have used RSVP-TE and that among these deployments, the vast majority did it for FRR reasons.

Our point is not to criticize the RSVP-TE protocol definition or minimize its merits. Like for LDP, there were good reasons 20 years ago for RSVP-TE and MPLS TE to have been defined the way they are. It is also clear that 20 years ago, RSVP-TE and MPLS-TE provided a major innovation to IP networks. At that time, there was no other bandwidth optimization solution. At that time, there was no other FRR solution. RSVP-TE and MPLS-TE introduced great benefits 20 years ago.

Our point is to look at its applicability in IP networks in 2016. Does it fit the needs of modern IP networks?

In our opinion, RSVP-TE and the classic MPLS TE solution have been defined to replicate FR/ATM in IP. The objective was to create circuits whose state would be signaled hop-by-hop along the circuit path. Bandwidth would be booked hop-by-hop. Each hop’s state would be updated. The available bandwidth of each link would be flooded throughout the domain using IGP to enable distributed TE computation.

We believe that these design goals are no longer consistent with the needs of modern IP networks.

First, RSVP-TE is not ECMP-friendly. This is a fundamental issue as the basic property of modern IP networks is to offer multiple paths from a source to a destination. This ECMP-nature is fundamental to spread traffic along multiple paths to add capacity as required and for redundancy reasons.

Second, to accurately book the used bandwidth, RSVP-TE requires all the IP traffic to run within so-called “RSVP-TE tunnels”. This leads to much complexity and lack of scale in practice.

Let’s illustrate this by analyzing the most frequent status of a network: i.e. a correctly capacity-planned network.

Such a network has enough capacity to accommodate without congestion a likely traffic volume under a set of likely independent failures. The traffic is routed according to the IGP shortest-path and enough capacity is present along these shortest-paths. This is the norm for the vast majority of SP and Enterprise networks either all of the times or at least for the vast majority of the times (this is controlled by the “likeliness” of the traffic volume and the failure scenarios). Tools such as Cisco WAN Automation Engine (WAE) Planning[2] are essential to correctly capacity plan a network.

HIGHLIGHT: Capacity Planning

If one wants to succeed in traffic engineering, one should first learn capacity planning. An analogy could be that if one wants to be healthy (SLA), one should focus on finding a life style (capacity planning process) which keeps the body and mind balanced such as to minimize when medicine (tactical traffic engineering) needs to be taken. We would advise to study WAE Planning.[2]

Clearly, in these conditions, traffic engineering to avoid congestion is not needed. It seems obvious to write it but as we will see further, this is not the case for an RSVP-TE network.

In the rare cases where the traffic is larger than expected or a non-expected failure occurs, congestion occurs and a traffic engineering solution may be needed. We write “may” because once again it depends on the capacity planning process.

Some operators might capacity plan the network via modeling such that these occurrences are so unlikely that the resulting congestion might be tolerated. This is a very frequent approach.

Some other operators may not tolerate even these rare congestions and then require a tactical traffic-engineering process.

A tactical traffic-engineering solution is a solution that is used only when needed.

To the contrary, the classic RSVP TE solution is an “always-on” solution. At any time (even when no congestion is occurring), all the traffic must be steered along circuits (RSVP-TE tunnels). This is required to correctly account the used bandwidth at any hop.

This is the reason for the infamous full-mesh of RSVP-TE tunnels. Full-mesh implies that there must be a tunnel from anywhere to anywhere on the network edge and that all traffic must ride on RSVP-TE tunnels. IP forwarding spoils accurate traffic statistics.

Hence, traffic can never utilize IGP derived ECMP paths and to hide the lack of ECMP in RSVP-TE, several tunnels have to be created between each source and destination (at least one per ECMP path).

Hence, while no traffic engineering is required in the most likely situation of an IP network, the RSVP-TE solution always requires N^2*K tunnels where N scales with the number of nodes in the network and K with the number of ECMP paths. While no traffic engineering is required in the most likely situation of an IP network, the classical MPLS TE solution always requires all the IP traffic to not be switched as IP, but as MPLS TE circuits.

The consequence of this "full-mesh" is lots of operational complexity and limited scale, most of the time, without any gain. Indeed, most of the times, all these tunnels follow the IGP shortest-path as the network is correctly capacity planned and no traffic engineering is required.

This is largely suboptimal. An analogy would be that one needs to wear his raincoat and boots every day while it rains only a few days a year.

RSVP operational complexity

"More than 15 years ago, when DT's IP/MPLS multi-service network design was implemented, everyone assumed that RSVP should be used as a traffic engineering technology in order to optimize the overall network efficiency. While the routers eventually proved that they can do RSVP, the operational experience was devastating: the effect of ECMP routing – even at that time – was completely underestimated, and mitigating the effects of parallel links with more and more TE tunnels made the overlay topology of explicit paths even more complicated.

Eventually we found that IGP metric tuning, which cannot optimize network efficiency as perfectly as explicit paths, still does a fair job in terms of network efficiency but at a much lower cost of operational complexity.

We continued to use RSVP for tactical cases of traffic-engineering for quite a while. We merge with a network that used RSVP for the sake of Fast-Reroute. But finally we managed to fulfill all requirements of efficiency, Fast-Reroute and disjoint paths just with proper IGP metric tuning, LFA based IP-FRR, and maintaining a suitable topology at the transport layer. Removing RSVP from the network design – although technically working – was considered a great advantage from all sides."

— Martin Horneffer

Let's remember the two origins (in our opinion) of the classical RSVP-TE complexity and scale problem: the modeling as a reference to ATM/FR circuit; the decision to optimize bandwidth in a distributed manner instead of a centralized one.

In the early 2000, Thomas Telkamp was managing the worldwide GLOBAL CROSSING backbone from Amsterdam, the Netherlands. This was one of the first RSVP-TE deployment and likely the biggest at that time. I had the chance to work directly with him and learned the three following concepts through the experience.

1. the always-on RSVP-TE full-mesh model is way too complex because it creates continuous pain for no gain as, in most of the cases, the network is just fine routing along the IGP shortest-path. A tactical TE approach is more appealing. Remember the analogy of the raincoat. One should only need to wear the raincoat when it actually rains.
2. ECMP is key to IP. A traffic engineering approach for IP should natively support ECMP
3. for real networks, routing convergence has more impact on SLA than bandwidth optimization

Let’s illustrate the third learning.

Back in the early 2000, GLOBAL CROSSING was operating one of the first VoIP service. During any network failure, the connectivity was lost for several 10’s of seconds waiting for the network (IGP plus full-mesh of RSVP-TE tunnels) to converge.

Considering that the network was correctly capacity planned for most expected failures, and that independent failures occur very often on a large network, it is easy to understand that the SLA impact of slow routing convergence is much more serious than the impact due to congestion.

While everyone was focusing at that time on QoS and TE (the rare congestion problem), a very important problem was left without attention: routing convergence.

Thanks to Thomas and the GLOBAL CROSSING experience, I then started the “Fast Convergence” project.

In 6 years, we improved IS-IS/OSPF convergence in a reference worldwide network from 9.5 second to under 200msec. This involved considerable optimization in all parts of the routing system from the IS-IS/OSPF process to the router’s linecard HW FIB organization including the RIB, LDP, LSD (the MPLS RIB), BCDL (the bus from the route processor to the linecard) and the FIB process. This involved considerable lab characterization either to monitor our progress towards our “sub-200msec” target or to spot the next bottleneck.

In parallel to this fast IS-IS/OSPF convergence, we also researched on an IP-based automated FRR for sub-50msec protection.

As we noted earlier, we knew that RSVP-TE deployment was rare (10%) and that most of these deployments were not motivated by BW optimization but rather by FRR. So, if we found an IP-optimized FRR that was simpler to operate, we knew that this would attract a lot of operator interest.

We started that research in 2001 timeframe at the cafeteria of the Brussels Cisco office. This was the "water flow" discussion. If the course of a river through a valley gets blocked, it suffices to explicitly steer the water to the top of the correct hill and, from there, let the water flow naturally to its destination.

The intuition was fundamental to our "IPFRR" research: the shorter the explicit path, the better.

Contrary to RSVP-TE FRR, we never wanted to steer the packet around the failure and back at the other end of the failure. This model is ATM/FR "circuit" centric. We wanted a model that would be IP centric. We wanted to reroute around the failure as fast as possible such as to release the packet as soon as possible to a natural IP path

Releasing the packet as soon as possible to a natural IP path was the target of our IPFRR project.

Very quickly, we found Loop-Free Alternate (LFA, RFC 6571).

LFA allowed the IGP to pre-compute FRR backup paths for 75 to 90% of the IGP destinations (please refer to RFC 6571 for LFA applicability analysis and reports of solution coverage across real data sets). This solution received a lot of interest as it offered a much simpler FRR alternative than RSVP-TE.

Later on, we extended the IPFRR coverage to 95/99% with the introduction of Remote LFA (RLFA, RFC 7490).

This was sufficient for most operators and the deployment of RSVP-TE for sole FRR reasons stopped in favor of the much simpler LFA/RLFA alternative.

Still, two problems were remaining: the theoretical lack of a 100% guarantee of coverage and the possibility that the LFA backup/repair path be non-optimum (not along the post-convergence path).

We had done ample research on the matter and we knew that we could not deliver these properties without explicit routing.

> “In early 2000, RSVP-TE was the only solution available to provide fast-reroute. Implementing RSVP-TE for FRR in a network already running LDP for primary path brings additional complexity in the network design and operations as three protocols (IGP, LDP, RSVP-TE) will interact between each other: think about the sequence of protocols events when a single link fails in the network, when something goes wrong, troubleshooting is not so easy.
>
> The availability of IPFRR solutions was a great opportunity for network simplification by leveraging only one primary protocol (IGP) to provide FRR.
>
> Even if LFA/RLFA did not provide 100% guarantee of protection, the gain in simplicity is a good reason to use them: a simple network is usually more robust.”
>
> — *Stéphane Litkowski*

In parallel to the IGP fast convergence and the IPFRR research, we had a third related research: the search for a microloop avoidance solution[3]: i.e. a solution that would prevent packet drop due to inconsistent transient state between converging routers.

We found several theoretical solutions along that research but never one with enough coverage or with enough robustness for real-life deployment.

Let's stop for a moment here and summarize the key points we learned through these years of design, deployment and research:

- LDP was redundant and was creating unneeded complexity
- For the bandwidth optimization problem, the RSVP-TE full-mesh was modelled on ATM/FR "circuits" with a distributed optimization and as a result was too complex and not scalable. A tactical ECMP-aware IP optimized solution would be better.
- For the FRR problem, the RSVP-TE solution was modelled on SONET/SDH "circuit" and as a result was too suboptimal from a bandwidth and latency viewpoint and was too complex to operate. An ECMP-aware IP-optimized solution which would release the packet as soon as possible to its shortest-path was much better. Our IPFRR research led to LFA and RLFA. Most operators were satisfied with the 90-99% coverage. Still, the 100% coverage guarantee was missing. We knew that a form of explicit routing would be required to get that property.
- MPLS was perceived to be too complex to deploy and manage and most Enterprise network and some SP network operators stayed away from it
- microloop avoidance was an unsolved problem

Around spring 2012, one of the research projects I was managing was related to OAM in ECMP networks.

It was indeed difficult for operators to spot forwarding issues involving a subset of the ECMP paths to a destination.

As part of that work, we came up with a proposal to allocate MPLS labels for each outgoing interface of each router and then steer OAM probes over a deterministic path by pushing the appropriate label stack on the probe. At each hop, the top label would be popped and would indicate the outgoing interface to forward the packet.

The proposal was not very interesting for several reasons.

First, BFD was already largely deployed on a per link basis and hence issues impacting any traffic on a link were already detected. What we had to detect is the failure related to a specific FIB destination and this idea was missing the target.

Second, this idea would require too many labels as one label per link hop was required and the network diameter could be very large.

However, at that time, I was planning a trip to Rome and the following intuition came to me: when I go to Rome, from Brussels, I listen to the radio for traffic events. If I hear that the Gottardo tunnel is blocked then I mentally switch on the path to Geneva, and then from there to the path to Rome.

This intuition initiated the Segment Routing project: in our daily life, we do not plan our journeys turn by turn; instead we plan them as a very small number of shortest-path hops. Applying the intuition to real network, real traffic engineering problems in real networks would be solved by one or two intermediate shortest-paths, one or two segments in SR terminology.

The years of designing and deploying real technology in real networks had taught me that the simplest ideas prevail and that unneeded sophistication leads to a lot of cost and complexity.

Obviously, I knew that I would want to prove this in a scientific manner by analyzing real-life networks and applying traffic engineering problems to them. I knew we needed to confirm it scientifically… but I had seen enough networks that I felt the intuition was correct.

HIGHLIGHT: Few segments would be required to express an explicit path

When we explain a path to a friend, we do not describe it turn by turn but instead as a small number of shortest-path hops through intermediate cities.

Applying the intuition to real network: real traffic engineering problems would be solved by one or two intermediate shortest-paths, one or two segments in SR terminology.

While the theoretical bound scales with the network diameter, few segments would be required in practice.

This was the start of the Segment Routing project.

We would distribute labels in the routing protocol (i.e. the prefix segments) and then we would build an ECMP-aware IP-optimized explicit path solution based on stacking few of these labels (i.e. the segment list). Doing so, we would drastically simplify the MPLS control-plane by removing LDP and RSVP-TE while we would improve its scalability and functionality (tactical bandwidth engineering, IPFRR with 100% coverage, microloop avoidance, OAM). We will explain these concepts in detail in this book.

In fact, the intuition of the “path to Rome” also gave the first name for “SR”. “SR” originally meant “Strade Romane” which means in Italian the network of roads that were built by the Roman Empire. By combining any of these roads, the Romans could go from anywhere to anywhere within the Empire.

Figure 1-1: Clarence at the start of the “Appia” roman road in Rome. The network of roman roads

Later on, we turned SR for Segment Routing :-).

1.3 The SDN and OpenFlow Influences

In 2013, two fundamental papers were published at Sigcomm: SWAN (Software-driven WAN) from Microsoft[4] and B4 from Google[5].

While reading these excellent papers, it occurred to me that while I was agreeing on the ideas, I would have not implemented them with an OpenFlow concept as, in my opinion, it would not scale.

Indeed, I had spent several years improving the routing convergence speed and hence I knew that the problem was not so much in the control plane and the algorithmic computation of a path but much more in the transferring of the FIB updates from the routing processor down to the linecards and then the writing of the updates from the linecard CPU to the hardware forwarding logic.

To give an order of magnitude, while the routing processor would compute and update an entry in usec, the rest of the process (distribute to Linecards, update hardware forwarding) was in msec when we started the fast convergence project. It took many years of work to get that component down to 10's of usec.

Hence, by intuition I would have bet that the OpenFlow-driven approach would run in severe convergence problem: it would take way too much time for the centralized control-plane to send updates to the switches and have them install these updates in HW.

Hint to the reader: do some reverse engineering of the numbers published in the OpenFlow papers and realize how slow these systems were.

Nevertheless, reading these papers was essential to the SR project.

Using my "path to Rome" intuition, I thought that a much better solution would be to combine centralized optimization with a traffic-engineering solution based on a list of prefix segments.

The need for centralized optimization is clearly described in a paper of Urs Holzle, also at Google.[6] It details the drawbacks due to the distributed signaling mechanism at the heart of RSVP-TE: lack of optimality, lack of predictability and slow convergence.

In a distributed signaling solution, each router behaves like a kid sitting at a table where bandwidth is represented as a pile of candies at the center of the table. Each router vies for its bandwidth as a kid for its candy. This uncoordinated fight leads to lack of optimality (each kid is not ensured to get its preferred taste or what is required for his health condition), lack of predictability (no way to guess which kid will get what candy) and slow convergence (kids fighting for the same candy, retrying for others etc.).

HIGHLIGHT: Centralized Optimization

Read the paper of Urs Holzle[6] for an analysis of the centralized optimization benefits over RSVP-TE distributed signaling: optimality, predictability and convergence.

While the need of centralized optimization was clear and I agreed with these papers, the use of OpenFlow-like technique was, in my opinion, the issue. It would lead to far too many interactions between the centralized controller and the routers in the network.

The first reason is that the OpenFlow granularity is far too small: a per-flow entry on a per switch basis. To build a single flow through the network, all the routers along the path need to be programmed. Multiply this by tens of thousands or millions and this is clearly too much state.

The second reason is that every time the network changes, the controller needs to re-compute the flow placement and potentially update a large part of the flow entries throughout the network.

Instead, the intuition was to let the centralized controller use segments as Lego bricks in the Lego construction toys. The network would offer basic Lego blocks as ECMP-aware shortest path to destinations (Prefix Segments) and the controller would combine these Lego blocks to build any explicit path it would want (but with as few per-flow state as possible, in fact only one, at the source).

To route a demand from San Francisco (SFO) to New York City (NYC) via Denver (DEN), rather than programming all the routers along the path with the appropriate per-flow entry, the controller would only need to program one single state at the source of the demand (SFO). The state would consist of a source-routed path expressed as an ordered list of segments {DEN, NYC}. This is what we refer to as SR Traffic Engineering (SRTE).

"Segment Routing represents a major, evolutionary step forward for the design, control and operation of modern, large-scale, data-center or WAN networks. It provides for an unprecedented level of control over traffic without the concomitant state required by existing MPLS control plane technologies. That it can accomplish this by leveraging MPLS data plane technologies means that it can be introduced into existing networks without major disruption. This accords it a significant ease-of-deployability advantage over other SDN technologies. It is an exciting time to be a network provider because the future of Segment Routing holds so much potential for transforming networks and the services provided by them."

— *Steven Lin*

SRTE is hybrid centralized/distributed cooperation between the controller and the network. The network maintains the multi-hop ECMP-aware segments and the centralized controller combines them to form a source-routed path through the network. State is removed from the network. State is only present at the ingress to the network and then in the packet header itself.

{DEN, NYC} is called a segment list. DEN is the first segment. NYC is the last segment. Steering a demand through {DEN, NYC} means steering the demand along the ECMP-aware shortest-path to Denver and then along the ECMP-aware shortest-path to NYC.

Note the similarity with the initial intuition for SR: if the shortest-path to Rome is jammed, then the alternate path is {Geneva, Rome}. The human mind does not express a path as a turn-by-turn journey; instead it expresses it as a minimum number of subsequent ECMP-aware shortest-paths.

In the early days of SR, we used a baggage tag analogy to explain SR to non-technical audience, see Figure 1-2. Imagine one needs to send a Baggage to Berlin (TXL) from Seattle with transit in Mexico City (MEX) and Madrid (MAD). Clearly, the transportation system does not associate a single ID to the baggage (flow ID) and then create circuit state in Mexico and Madrid to recognize the baggage ID and route accordingly. Instead, the transportation system scales better by appending a tag "Mexico then Madrid then Berlin" to the baggage at the source of the journey. This way, the transportation system is unaware of each individual baggage specifics along the journey. It only knows about a few thousands of airport codes (Prefix Segments) and routes the baggage from airport to airport according to the baggage tag. Segment Routing does exactly the same: a Prefix Segment acts as an airport code; the Segment List in the packet header acts as a tag on the baggage; the source knows the specifics of the packet and encodes it as a Segment List in the packet header; the rest of the network is unaware of that specific flow and only knows about a few hundreds or thousands of Prefix Segments.

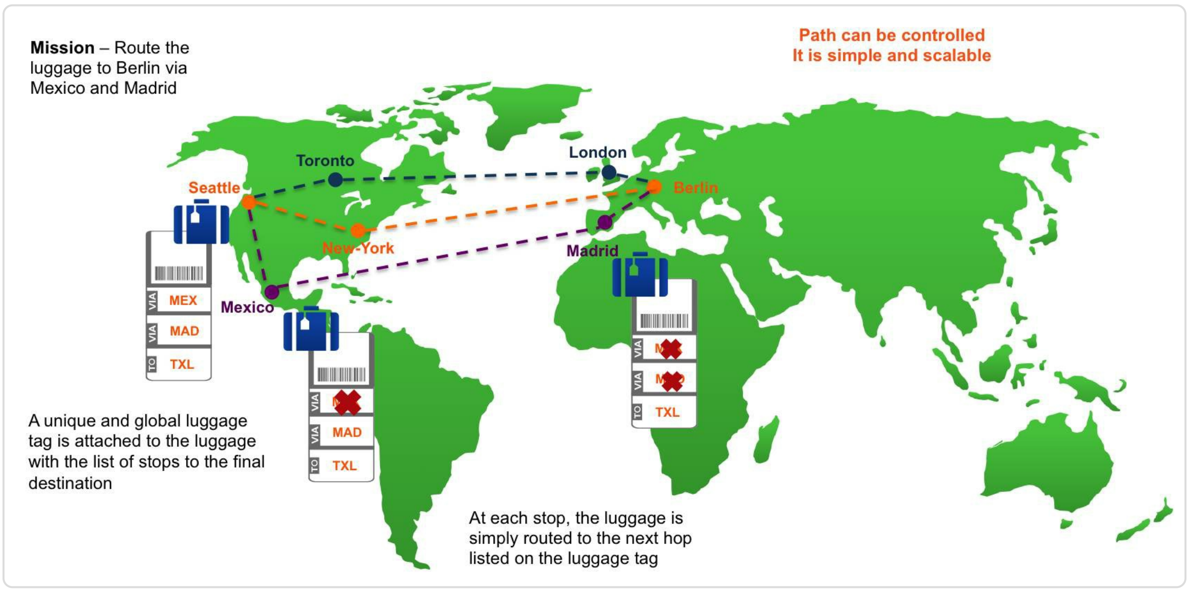

Mission – Route the luggage to Berlin via Mexico and Madrid
Path can be controlled
It is simple and scalable
Toronto
London
Seattle
Berlin
New-York
Madrid
Mexico
VIA MEX
VIA MAD
TO TXL
VIA
VIA MAD
TO TXL
VIA
VIA
TO TXL
A unique and global luggage tag is attached to the luggage with the list of stops to the final destination
At each stop, the luggage is simply routed to the next hop listed on the luggage tag

Figure 1-2: Baggage tag analogy to explain SR to non-technical audience

HIGHLIGHT: Combining segments to form an explicit path.

Hybrid coupling of centralized optimization with distributed intelligence

The network would offer basic Lego bricks as ECMP-aware shortest path to destinations (Prefix Segments). The distributed intelligence would ensure these segments are always available (IGP convergence, IP FRR).

> The controller would acquire the entire topology state (with the segments supported by the network) and would use centralized optimization to express traffic-engineering policies as a combination of segments. It would consider segments as a Lego bricks. It would combine them in the way it wants, to express the policy it wants.
>
> The controller would scale better thanks to the source-routing concept: it would only need to maintain state at the source, not throughout the infrastructure.

The network would keep the right level of distributed intelligence (IS-IS and OSPF distributing the so-called Prefix and Node Segments) and the controller would express any path it desires as a list of segments. The controller's programming job would be drastically reduced because state would only need to be programmed at the source and not all along the flow path. The controller scaling would also be much better thanks to the reliance on the IGP intelligence. Indeed, in real life, the underlying distributed intelligence can be trusted to adapt correctly to the topology changes reducing the need for the controller to react, re-compute and update the segment lists. For example, the inherent IGP support for an FRR solution ensures that the connectivity to prefix segments is preserved and hence the connectivity along a list of segments is preserved. Upon failure, the controller can leverage this IGP FRR help to dampen its reaction and hence scale better.

This is why we say that Segment Routing is hybrid centralized/distributed optimization architecture.

We marry distributed intelligence (shortest-path, FRR, microloop avoidance) with centralized optimization for a policy objective like latency, disjoint-ness, avoidance and bandwidth.

In SR, we would tackle the bandwidth engineering problem with a centralized controller in a tactical manner: the controller would monitor the network and the requirements of applications and would push SRTE policies only when required. This would give us better optimality and predictability. We would scale the controller by expressing traffic engineering policies with list of segments and hence would only need per-demand state at the source.

Also our intuition was that this solution would also converge faster.

Why? Because the comparison has to be made with an N^2*K full-mesh of RSVP-TE tunnels that vie for bandwidth every time the topology changes. It was well-known that operators had seen convergence times as slow as 15minutes. In comparison, our approach would only have to compute SRTE policies in the rare cases when congestion occurs. Few states would need to be programmed thanks to the source-routed nature. By combining sufficient centralized compute performance with our scaled source-routing paradigm, likely we would reduce the convergence time. This will be studied in Part II of the book.

Figure 1-3: John Leddy highlighting the importance to keep the network stateless and hence the role of for SDN done right

On SDN and the role of Segment Routing

“SR is SDN done right!”

— John Leddy

The provisioning of such tactical SRTE policies can also be done directly on the edge router in the network as a first transition, without the need for a controller. However, the benefits of bringing in the centralized intelligence are obvious as described above, and enable the transition to an SDN architecture. We will see in this book how basic Segment Routing addresses the requirements for which operators used to turn to RSVP-TE and in the next book we will introduce real traffic engineering use-cases at a scale that was challenging if not impossible until now.

Aside this influence, SDN proved fundamental to SR, it simply allowed it to exist: the applicability of SR to SDN provided us with the necessary tailwind to push through our proposal against the dominance of the classic MPLS control-plane.

Over the last three years, seeing the numerous design and deployments, we can say that SR has become the de-facto network architecture for SDN. Watch for example the SR analysis for SWAN.[7]

1.4 100% Coverage for IPFRR and Optimum Repai Path

Now that we had a source-routed explicit solution, it was straightforward to close our IPFRR research.

LFA and RLFA had many advantages but only gave us 99% coverage and were not always selecting the optimum path.

Thanks to SR, we could easily solve both problems.

The point of local repair (PLR) would pre-compute the post-convergence path to a destination, assuming its primary link, node or SRLG fails. The PLR would express this explicit path as source-routed list of segments. The PLR would install this backup/repair path on a per destination basis in the data plane. The PLR would detect the failure in sub-10msec and would trigger the backup/repair paths in a prefix-independent manner.

This solution would give us 100% coverage in any topology. We would call it "Topology-Independent LFA" (TI-LFA). It would also ensure optimality of the backup path (post-convergence path on a per-destination basis).

We will explain all the details of TI-LFA in this book.

HIGHLIGHT: TI-LFA benefits

- Sub-50msec link, node and SRLG protection
- 100% coverage.
- Simple to operate and understand
- Automatically computed by the IGP, no other protocol required

- No state created outside the protecting state at the PLR
- Optimum: the backup path follows the post-convergence path
- Incremental deployment
- Also applies to IP and LDP traffic

1.5 Other Benefits

The following problems in IP/MPLS networks did also motivate our SR research:

- IP had no native explicit routing capability.
  - It was clear that with explicit routing capability, IPv6 would have a central role to play in future infrastructure and service network programming.
  - SRv6 is playing a key role here. We will detail this in Part II of the book.
- RSVP-TE did not scale inter-domain for basic services such as latency and disjoint-ness.
  - In a previous section, we analyzed the drawbacks of the RSVP-TE solution for the bandwidth optimization problem. RSVP-TE has another issue: for latency or disjoint-ness services, it did not scale across domains. The very nature of modern IP networks is to be multi-domain (e.g. data-center (DC), metro, core).
  - SR is now recognized as a scalable end-to-end policy-aware IP architecture that spans DC, metro and backbone. This is an important concept which we will detail in this book (the scalability and base design) but as well in Part II (the TE specific aspects).
- MPLS did not make any inroad in the Data Center (DC)
  - We felt that while MPLS data plane had a role to play in the DC, this could not be realized due to inherent complexities of the classic MPLS control plane.
  - SR MPLS is now a reality in the DC. We will document this in the

    BGP Prefix-SID section and the DC use-case.

- OAM was fairly limited
    - Operators were reporting real difficulty to detect forwarding issues in IP network, especially when the issue is associated to a specific FIB entry (destination based) potentially for a specific combination of ingress port at the faulty node and ECMP hash.
    - SR has delivered solutions here which will be detailed in the OAM section of this book.
- Traffic Matrices were important but were actually unavailable to most operators.
    - Traffic matrices are the key input to the capacity planning analysis that is one of the most important processes within operator. Most operators did not have any traffic matrices. They were basically too complex to be built with classic IP/MPLS architecture.
    - SR has delivered an automated traffic matrix collection solution. We will detail it in the next book.

1.6 Team

David Ward, SVP and Chief Architect for Cisco Engineering, was essential to realize the opportunity with SR and approved the project in September 2012.

We have created the momentum by keeping focus on the execution and delivering an impressive list of functionality and use-cases: IS-IS and OSPF support for SR, BGP support for SR, the SR/LDP seamless interworking, the TI-LFA 100% FRR solution, microloop avoidance, distributed SRTE, centralized SRTE, the egress peer traffic engineering use-case etc.

Ahmed Bashandy was the first person I recruited for the SR project. I had worked with him during the Fast Convergence project and I knew him well. He had deep knowledge across the entire IOS-XR system (he had coded in IS-IS, BGP and FIB). This was essential for bringing up our first SR implementation as we had to touch all the parts of the system. Ahmed is fun to work with. This is essential. Finally, Ahmed is not slowed down by doubts. Many engineers would have spent hours asking why we would do it, what was the difference against MPLS classic, whether we had to first have IETF consensus, whether we had any chance to get something done in the end… all sorts of existential questions that are the best way to never do anything. This was certainly not going to be an issue with Ahmed and this was essential. We had no time to lose. Ahmed joined in December 2012 and we had a first public demo by March 2013.

Bertrand Duvivier was the second person to join the team. I needed someone to manage the relationship with marketing and engineering once we became public. I knew Bertrand since 1998. He is excellent at understanding a technology, the business benefits and managing the engineering and marketing relationships. Bertrand joined the team in February 2013. We come from the same region (Namur, Belgium), we speak the same language (French, with same accent and dialect), share the same culture hence it is very easy to understand each other.

Kris Michielsen was the third person to join. Again, I knew him from the Fast Convergence project. Kris had done all the Fast Convergence characterization. He is excellent at managing lab characterization, thoroughly analyzing the origins of bottlenecks, creating training and transferring information to the field and operators. We knew each other since 1996.

Stefano Previdi was the fourth person to join. I knew him since I started at Cisco in June 1996. We had done all the MPLS deployments together and then IPFRR. Stefano would focus on the IETF and we will talk more about this later.

Aside the technical expertise, we were all in the same time-zone, we knew each other for a long time, we could understand each other well and we had fun working together. These are essential ingredients.

Later on, Siva Sivabalan and Ketan Talaulikar would join us. Siva would lead the SRTE project and Ketan would contribute to the IGP deployment.

Operators have played a fundamental role in the SR project.

In October 2012, at the annual Cisco Network Architecture Group (NAG) conference (we invite key operator architects for 2 days and have open discussion on any relevant subject of networking, without any marketing compromise), I presented the first session on SR, describing the improved simplicity, the improved functionality (FRR, SRTE), the improved scale (flow state only at the source) and the opportunity for SDN (hybrid centralized and distributed optimization). This session generated a lot of interest.

We immediately created a lead operator group and started defining SR.

John Leddy from Comcast, Stéphane Litkowski and Bruno Decraene from Orange, Daniel Voyer from Bell Canada, Martin Horneffer from DT, Rob Shakir then at BT, Josh George then at Google, Ebben Aries then at Facebook, Dmitry Afanasiev and Daniel Ginsburg at Yandex, Tim Laberge and Steven Lin then at Microsoft, Mohan Nanduri and Paul Mattes from Microsoft were among the most active in the group.

Later on, as the project evolved, many other engineers and operators joined the effort and we will have the opportunity to introduce their viewpoints throughout this book.

Again, most of us were in the same time-zone; we knew each other from previous projects. It was easy to understand each other and we had the same focus. This generated a lot of excellent discussions that shaped most of the technology and use-cases illustrated in this book.

Within a few months of defining this technology, we received a formal letter of intent to deploy from an operator. This was a fantastic proof for the business interest and this helped us fund our first phase of the project.

Ravi Chandra proved essential to execute our project beyond its first phase. Ravi was leading the IOS-XR, IOS-XE and NX-OS software at Cisco. He very quickly understood the SR opportunity and funded it as a portfolio-wide program. We could then really tackle all the markets interested in SR (hyper-scale WEB operators, SP and Enterprise) and all the network segments (DC, metro/aggregation, edge, backbone).

> “Segment Routing is a core innovation that I believe will change how we do networking like some of the earlier core technologies like MPLS. Having the ability to Source Routing at scale will be an invaluable tool in many different use cases in the future.”
>
> *— Ravi Chandra*
> *SVP Cisco Systems Core Software Group*

1.7 Keeping Things Simple

More than anything, we believe in keeping things simple.

This means that we do not like technology for the sake of technology. If we can avoid technology, we avoid technology. If we can state a reasonable design guideline (that is correct and fair for real networks) such that the problem becomes much simpler and we need less technology to solve it, then we are not shy of making the assumption that such guideline will be met.

With SR, we try to package complex technology and algorithms in a manner that is simple to use and operate. We favor automated behaviors. We try to avoid parameters, options and tuning. TI-LFA, microloop avoidance, centralized SRTE optimization (Sigcomm 2015 paper[8]), and distributed SRTE optimization are such examples.

Simplicity

“The old rule to make things "as simple as possible, but not simpler" is always good advice for network design. In other words it says that the parameter complexity should be minimized.

However, whenever you look closer at a specific problem, you will usually find that complexity is a rather multi-dimensional parameter. Whenever you minimize one dimension, you will increase another. For example, in order to provide sets of disjoint paths in an IP backbone, you can either ask *operations* to use complicated technology and operational procedures to satisfy this requirement. Or you can ask *planning* to provide a suitable topology with strict requirements on the transport layer.

The art of good network design is, in my eyes, based on a broad overview of possible dimensions of complexity, good knowledge of the cost related to each dimension, and wise choice of the trade-offs that come with specific design options.

I am convinced that segment routing offers a lot of very interesting network design options that help to reduce overall network cost and increase its robustness.”

— *Martin Horneffer*

The point of keeping things simple is to choose well where to put intelligence to solve a real problem, how to package the intelligence to simplify its operation and where to simplify the problem by an appropriate design approach.

1.8 Standardization and Multi-Vendor Consensus

When we created the lead-operator group, we pledged three characteristics of our joint work: (1) transparency, (2) committed to standards, (3) committed to multi-vendor consensus.

The transparency meant that we would define the technology together; we would update the lead operator group with progress, issues and challenges.

Our commitment to standardization meant that we would release all the necessary information to the IETF and would ensure that our implementation will be entirely in-line with the released information. We clearly understood that this was essential to our operators and hence it was essential to us.

Clearly, getting our ideas through the IETF was a fantastic challenge. We were challenging 20-years of classic MPLS control-plane and were promoting concepts (global label) that were seen as a non-starter by some prominent part of the IETF community.

Our strategy was the following:

1. be positive. We knew that we would be attacked in cynical ways. The team was instructed to never reply and never state anything negative or emotional
2. lead by the implementation. We had to take the risk of implementing what we were proposing. We had to demonstrate the benefits of the use-cases.

3. get the operators to voice their requirements. The operators who wanted to have SR knew that they would need to state their demands very clearly. The implementation and the demonstration of the use-cases would strengthen their argumentation
4. get other vendors to contribute. Alcatel (Wim Henderickx) and Ericsson (Jeff Tantsura) quickly understood the SR benefits and joined the project. More or less a year after our initial implementation, we (i.e. Cisco) could demonstrate interoperability with Alcatel and Ericsson and it was then clear that the multi-vendor consensus would be obtained. Shortly after, Huawei also joined the effort.
5. shield the team from this standardization activity while handling the process with the highest quality

Stefano joined the team to handle this last point. He would lead and coordinate all our IETF SR involvement on behalf of the team. The team would contribute by writing text but the team would rarely enter in list discussion and argumentation. Stefano would handle it. This was essential to get the highest quality of the argumentation, the consistency but as well to shield the team from the emotional implication of the negotiation. While we were handling the most difficult steps of the IETF process and negotiation, the rest of the engineers were focused on coding and releasing functionality to our lead operators. This was key to preserve our implementation velocity, which was a key pillar of the strategy.

Standardization and Multi-Vendor Focus for SR team

"Operators manage multi-vendors networks. Standardization and interoperability is thus essential, especially for routing related subjects.

It would have been easier and faster for the SR team to work alone and create a proprietary technology as big vendors sometimes do.

Yet, Clarence understood the point, committed to work with all vendors and did make it through the whole industry.”

— Bruno Decraene

1.9 Global Label

Clearly, SR was conceived with global labels in mind.

Since global labels have been discussed in the IETF in the early days of MPLS, we knew that some members of the community would be strongly against this concept. Very early, we came up with the indexing solution in the SRGB[9] range. We thought it was a great idea because on one hand, operators could still use the technology with global labels (simply by ensuring that all the routers use the same SRGB range), while on the other hand, the technology was theoretically not defining global labels (but rather global indexes) and hence would more easily pass the IETF process.

We presented this idea to the lead operator group in February 2013 and they rejected it. Main concern has been that they wanted global labels, they wanted the ease of operation of global labels and hence they did not want this indexing operation.

We noted their preference and when we released the proposal publicly in March 2013, our first IETF draft and our first implementation were using global labels without index.

As expected, global labels created lots of emotions… To get our multi-vendor consensus, we invited several vendors to Rome for a two-day discussion (see Figure 1-4). We always meet in Rome when an important discussion must occur. This is a habit that we took during the IPFRR project.

Figure 1-4: Note the bar "Dolce Roma" ("Sweet Rome") behind the team :-). From left to right, Stefanc Previdi, Peter Psenak, Wim Henderickx, Clarence Filsfils and Hannes Gredler

During the first day of the meeting, we could resolve all issues but one: Juniper and Hannes were demanding that we re-introduce the index in the proposal. Hannes' main concern was that in some contexts with legacy hardware-constrained systems, it may not be possible to come up with a single contiguous SRGB block across vendors.

At the end of the first day, we then organized a call with all the lead operators, explained the status and advised to re-introduce the index. Multi-vendor consensus and standardization was essential and operators could still get the global label behavior by ensuring that the SRGB be the same on all nodes.

Operators were not happy as they really wanted the simplicity of the global labels but in the end agreed.

On the second day, we updated the architecture, IS-IS, OSPF and use-case drafts to reflect the changes and from then on, we had our multi-vendor consensus.

This story is extremely important as it shows that SR (for MPLS) was really designed and desired with global labels and that operators should always design and deploy SR with the same SRGB across all their routers. This is the simplest way to use SR and the best way to collect all its benefits.

Sure, diverse SRGB is supported by the implementation and the technology… but really understand that it is supported mostly for the rare interoperability reasons[10] in case no common SRGB range could be found in deployments involving legacy hardware constrained systems. People really would like to use as per the original idea of global labels.

During the call with the operators, as they were very reluctant to accept the index, I committed that the Cisco implementation would allow for global and index configuration both in the configuration and in the show command. Indeed, by preserving the look and feel of the global label in the configuration and in the show command, by designing with the same SRGB across all the routers, an operator would have what they want: global label. They would never have to deal with index.

This section is a good example for the use of “opinion” textboxes. Without this historical explanation, one could wonder why we have two ways to configure prefix segments in MPLS. This is the reason.

The use of the SRGB, the index, global and local labels are described in detail in this book.

1.10 SR MPLS

It is straightforward to apply the SR architecture to the MPLS data plane.

A segment is a label. A list of segments is a stack of labels. The active segment is the top of the stack. When a segment is completed, the related label is popped. When an SR policy is applied on a packet, the related label stack is pushed on the packet.

The SR architecture reuses the MPLS data plane without any change. Existing infrastructure only requires a software upgrade to enable the SR control-plane.

Operators were complaining about the lack of scale, functionality and the inherent complexity of the classic MPLS control plane. The MPLS data plane was mature and very well deployed. For these reasons, our first priority has been to apply SR to the MPLS data plane. This was the main focus of our first three years of the SR project and this is also the focus of this book.

It is important to remind that SR MPLS applies to IPv4 and IPv6.

As part of the initial focus on SR MPLS, it was essential to devise a solution to seamlessly deploy SR in existing MPLS network. The SR team and the lead operator group dedicated much of the initial effort to this problem. A great solution was found to allow SR and LDP brownfield networks to seamlessly co-exist or, better, inter-work.

This SR/LDP seamless solution is described in detail in this book.

1. 11 SRv6

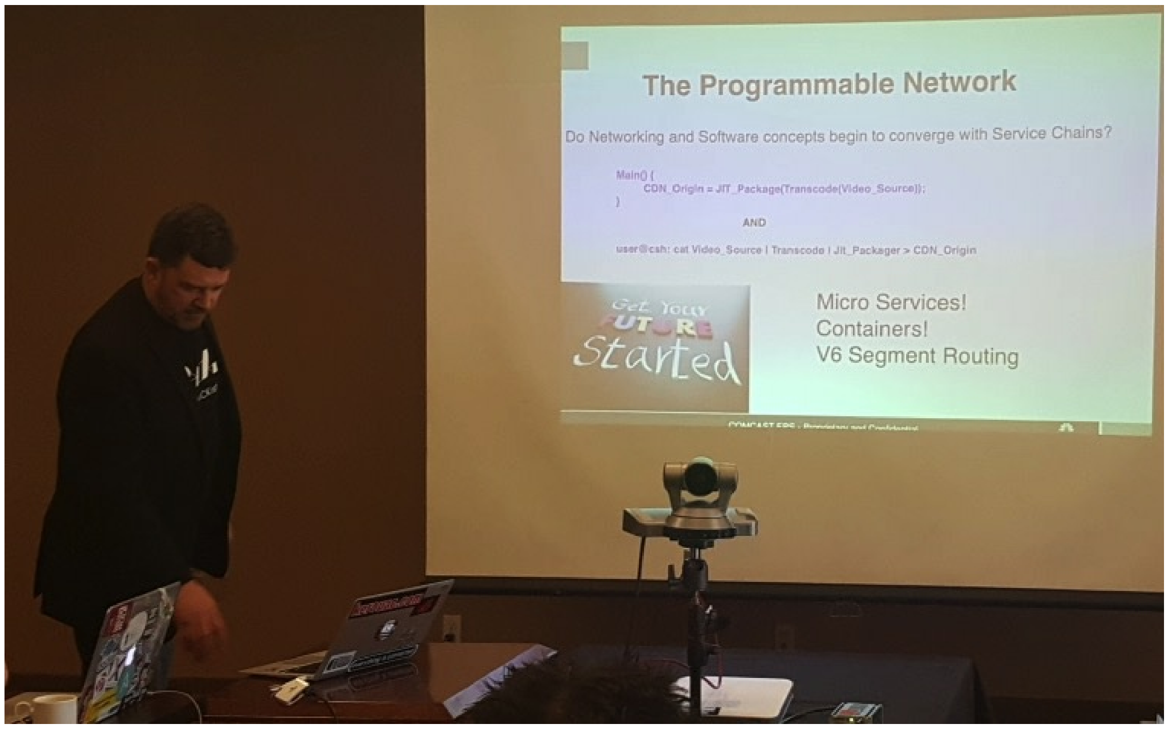
The SR architecture has been thought since day one for its application to the IPv6 data plane. This is referred to as “SRv6”.

All the research we did on automated TI-LFA FRR, microloop avoidance, distributed traffic engineering, centralized traffic engineering etc. is directly applicable to SRv6.

We believe that SRv6 plays a fundamental role to the value of IPv6 and will significantly influence all future IP infrastructure deployments either in the DC, the large-scale aggregation, in the backbone.

SRv6’s influence will expand beyond the infrastructure layer. An IPv6 address can identify any object, any content or any function applied to an object or a piece of content. SRv6 could offer formidable opportunities for chaining micro-services in a distributed architecture or for content networking.

We would recommend reading the paper of John Schanz from Comcast[11], watching the session of John Leddy on Comcast’s SRv6-based Smarter Network concept[12] and watching the demonstration of the “Spray” end-to-end SRv6 use-case[13] to understand the potential of SRv6.



The Programmable Network

Do Networking and Software concepts begin to converge with Service Chains?

```
Main() {
      CDN_Origin = JIT_Package(Transcode(Video_Source));
}

                    AND

user@csh: cat Video_Source | Transcode | Jit_Packager > CDN_Origin
```

Micro Services!
Containers!
V6 Segment Routing

Figure 1-5: John Leddy presenting on the "Smarter Network" concept and highlighting the SRv6 role

John Leddy played a key role in highlighting the SRv6 potential for drastically enhancing the application interaction with the network. We will focus on the SRv6 technology and use-cases in the second book. This first book introduces the Segment Routing architecture by first addressing the simpler use-cases related to the MPLS data plane and the MPLS infrastructure from DC to aggregation to backbone.

In this part of the book, we will only provide a small introduction to SRv6. We will dedicate much more content in a following book.

1.12 Industry Benefits

We have seen SR being applied to the hyper-scale WEB, SP and Enterprise markets. We have seen use-cases in the DC, in the metro/aggregation and in the WAN. We have seen (many) use-cases for end-to-end policy-aware architecture from the server in the DC through the metro and the backbone.

We believe that the SR benefits are the following:

- simplification of the control-plane (LDP and RSVP-TE removed, LDP/IGP synchronization removed)
- topology-independent IP-optimal 50msec FRR (TI-LFA)
- microloop avoidance
- support for tactical traffic engineering (explicit path encoded as a segment list)
- centralized optimization benefits (optimality, predictability, convergence)
- scalability (per-flow state is only at the source, not throughout the infrastructure)
- seamless deployment in existing networks (applies equally for SR-MPLS and SRv6)
- de-facto architecture for SDN
- standardized
- multi-vendor consensus
- strong requirement from operators

- defined closely with operators for real use-cases
- solving unsolved problems (TI-LFA, microloop, inter-domain disjointness/latency policies…)
- cost optimization (through improved capacity planning with tactical traffic engineering)
- an IPv6 segment can identify any object, any content, or any function applied to an object. This will likely extend SR's impact well beyond the infrastructure use-cases.

Most of these benefits will be described in this book. The TE and SRv6 benefits will be detailed in the next book.

1. 13 References

- [draft-francois-rtgwg-segment-routing-uloop] Francois, P., Filsfils, C., Bashandy, A., Litkowski, S. , "Loop avoidance using Segment Routing", draft-francois-rtgwg-segment-routing-uloop (work in progress), June 2016, https://datatracker.ietf.org/doc/draft-francois-rtgwg-segment-routing-uloop
- [RFC5443] Jork, M., Atlas, A., and L. Fang, "LDP IGP Synchronization", RFC 5443, DOI 10.17487/RFC5443, March 2009, https://datatracker.ietf.org/doc/rfc5443.
- [RFC6138] Kini, S., Ed., and W. Lu, Ed., "LDP IGP Synchronization for Broadcast Networks", RFC 6138, DOI 10.17487/RFC6138, February 2011, https://datatracker.ietf.org/doc/rfc6138.
- [RFC6571] Filsfils, C., Ed., Francois, P., Ed., Shand, M., Decraene, B., Uttaro, J., Leymann, N., and M. Horneffer, "Loop-Free Alternate (LFA) Applicability in Service Provider (SP) Networks", RFC 6571, DOI 10.17487/RFC6571, June 2012, https://datatracker.ietf.org/doc/rfc6571.
- [RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, https://datatracker.ietf.org/doc/rfc7490.
- [RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, https://datatracker.ietf.org/doc/rfc7490.

[1] Note that this entire chapter should be considered subjective

[2] Cisco WAN Automation Engine (WAE), http://www.cisco.com/c/en/us/products/routers/wae-planning/index.html and http://www.cisco.com/c/en/us/support/routers/wae-planning/model.html

[3] draft-francois-rtgwg-segment-routing-uloop, https://datatracker.ietf.org/doc/draft-francois-rtgwg-segment-routing-uloop

[4] http://conferences.sigcomm.org/sigcomm/2013/papers/sigcomm/p15.pdf

[5] http://conferences.sigcomm.org/sigcomm/2013/papers/sigcomm/p3.pdf

[6] http://www.opennetsummit.org/archives/apr12/hoelzle-tue-openflow.pdf

[7] Paul Mattes, "Traffic Engineering in a Large Network with Segment Routing ", Tech Field Day, https://www.youtube.com/watch?v=CDtoPGCZu3Y

[8] http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p15.pdf

[9] Segment Routing Global Block, see chapter 4, "Management of the SRGB"

[10] In extreme/rare cases, this could help to migrate some hardware constrained legacy systems, but one should be very careful to limit this usage to a transient migration. To the best of our knowledge, all Cisco platforms support the homogenous SRGB design guideline at the base of SR MPLS.

[11] John D. Schanz, “How IPv6 lays the foundation for a smarter network”, http://www.networkworld.com/article/3088322/internet/how-ipv6-lays-the-foundation-for-a-smarter-network.html

[12] John Leddy, “Comcast and The Smarter Network with John Leddy”, Tech Field Day, https://www.youtube.com/watch?v=GQkVpfgjiJ0

[13] Jose Liste, “Cisco Segment Routing Multicast Use Case Demo with Jose Liste”, Tech Field Day, https://www.youtube.com/watch?v=W-q4T-vN0Q4

2 FUNDAMENTALS OF SEGMENT ROUTING

2.1 What is Segment Routing?

The Segment Routing (SR) architecture is based on the *source routing* paradigm. A node (any router, host, or a device in general) chooses a path and steers a packet through the network via that path by inserting an ordered list of *segments* in the packet header, instructing how nodes that receive the packet should process this packet. A segment can represent any kind of instruction - topological, service-based, context, etc.

Because the instructions are encoded in the packet header, the nodes in the network simply have to execute these instructions when they receive the packet. The nodes on that path do not have to maintain state information for all possible flows that are traversing them ("the state is in the packet").

The source node can control the steering of packets at the level of individual flows by adding the proper instructions, or *segments*, in the packet headers. Since nodes, other than the source node, do not need to store and maintain any per-flow state, the traffic steering decisions only impact the source nodes. This way, SR offers advanced traffic steering capabilities in IP and MPLS networks while maintaining scalability both in the data plane and control plane.

2.2 Concept of Segments

The SR architecture does not assume a specific data plane implementation. This chapter first presents an abstract Segment Routing model that then can be applied to specific data plane implementations.

Subsequent chapters will review the abstract concepts defined in this chapter with a pragmatic description of the MPLS implementation, examples and use-cases.

2.2.1 Segments and Segment Identifiers

A Segment is an instruction that a node executes on the incoming packet, which carries the instruction in its header. Examples of instructions are: *forward packet according to the shortest path to destination, forward packet through a specific interface*, *deliver packet to a given application/service instance*, etc.

A Segment Identifier (SID) identifies a Segment. The format of a SID depends on the implementation. Examples of SID formats are: an MPLS label, an index value in an MPLS label space, an IPv6 address. The terms "*segment*" and "*SID*" are used interchangeably in this book.

2.2.2 Combining Segments

Segments are basic building blocks to build paths through the network. In the simplest application, these building blocks can be used individually. A single segment can steer a packet through the network, such as a segment with the instruction *forward packet according to shortest path to destination Node1*. Other applications can combine these building blocks to achieve the requirements of the application data flow. By combining segments in an ordered list, packets can be steered on any path through the network, regardless of shortest path, domain boundaries, routing protocol, etc. Or by combining segments, packets can be steered through a chain of services. Such an ordered list of segments is called a Segment List or SID List. Each entry in the SID List is an instruction to complete one section or segment of the whole path, therefore the term “segment” in Segment Routing.

2.2.3 Segment List Operations

A Segment Routed packet carries a Segment List or SID List, in its header. The instruction that is currently being executed on the packet is the instruction of the *Active Segment*. If the Active Segment of a packet has the instruction *forward packet according to shortest path to destination Node1*, then each node that receives that packet forwards the packet on the shortest path to Node1.

Besides executing the instructions encoded in the Segment List, the nodes also maintain the Segment List itself. Three basic Segment List operations are defined for this purpose:

PUSH

insert segment(s) at the head of the Segment List and set the first segment as the Active Segment. If the packet did not have a Segment List in its header, it is inserted first.

CONTINUE

the Active Segment is not completed and hence remains active.

NEXT

the Active Segment is completed; the next segment in the Segment List becomes the Active Segment.

These Segment List operations are mapped to the actual data plane operations done on the packet header. Currently SR has been defined for the MPLS and IPv6 data planes.

2.2.4 Global and Local Segments

The term “Global” in this book means “within the SR Domain”. The SR Domain is the set of nodes participating in the SR model.

2.2.4.1 Global Segment

All SR enabled nodes in the SR Domain support the instruction that is associated with a Global Segment and each node in the SR Domain installs the instruction of a Global Segment in its forwarding table. An example of a Global Segment is a segment with the instruction *forward packet according to shortest path to destination Node1*. Since it is a Global Segment, each node in the SR Domain knows how to forward the packet on the shortest path to Node1.

2.2.4.2 Local Segment

Only the node that originates a Local Segment supports the instruction that is associated with that Local Segment and only the node that originated the Local Segment installs the associated instruction in its forwarding table. An example of a Local Segment is a segment advertised by Node1 with the instruction *forward packet on interface to Node2*. Although other nodes can have an interface to Node2, the intention is to steer the packet on Node1’s interface to Node2.

To correctly execute the intended instruction of a Local Segment, the packet first has to be steered to the node that can correctly execute this instruction. In our example we want Node1 to execute the instruction, therefore a Global Segment with instruction *forward packet according to shortest path to destination Node1* can precede the Local Segment. The packet will then be steered first to Node1 and then on Node1's interface to Node2.

The local semantic of a Local Segment does not mean that nodes in the network do not know about other nodes' Local Segments. In order to be able to use Local Segments, other nodes need to know about their existence and meaning.

2.3 Segment Routing Control Plane

The SR architecture does not assume a specific control plane implementation. Although static configuration of segment instructions on the nodes in the network would be possible in theory, a routing protocol is generally used to distribute the segment information in the network. The SR control plane is currently specified for the link-state Interior Gateway Protocols (IGPs) IS-IS and OSPF, and for the Border Gateway Protocol (BGP).

It does not come as a surprise that the segments distributed by IGP are called *IGP Segments* and those distributed by BGP are called *BGP Segments*.

The types of segments described in this section are not meant to be an exhaustive list, but they are the foundational building blocks. Other types of segments exist and likely even more segment types will be created in the future.

2.3.1 IGP Segments

An IGP Segment, or IGP SID, is a type of segment that is attached to a piece of information advertised by a link-state IGP, such as an IGP prefix or an IGP adjacency. The inclusion of the segments in IGP advertisements requires some limited extensions of the link-state IGPs. Both IS-IS and OSPF are extended to support the distribution of segment IDs. The details of these protocol extensions are explained in chapter 5, "Segment Routing IGP Control Plane".

Two main (sub-)types of IGP segments can be distinguished: IGP Prefix Segments and IGP Adjacency Segments. These segment types are respectively associated with an IGP prefix and an IGP adjacency. These two IGP Segment types are the basic building blocks that allow constructing any topological path through an IGP network by combining or stacking these building blocks.

IGP Segments are distributed using the link-state IGP advertisements. This means that all nodes in the IGP network receive the IGP Segments. This way the IGP provides a complete view of the network: each node in the network knows both the topology of the network and all the segments in the area. If the IGP network is organized in areas, then the nodes do not have a complete view of the remote area (i.e. the topology of nodes and links within it), since that is the principle of separating a network in areas. The IGP Segments follow the distribution of their corresponding link, node and prefix elements. As prefix reachability information is propagated between areas, their associated segment information is also propagated along with it. This way Segment Routing can also be used as a default transport method in multi-area and multi-level networks.

2.3.1.1 IGP Prefix Segment

The IGP Prefix Segment (or commonly abbreviated to just "Prefix Segment" or "Prefix-SID") is a Global Segment, advertised by IS-IS or OSPF, associated with a prefix advertised by this IGP. The instruction of an IGP Prefix Segment is "*steer traffic along the ECMP-aware shortest path to the prefix associated with this segment*". From anywhere in the network, packets with a Prefix Segment as Active Segment are forwarded on the ECMP-aware shortest-path towards the node advertising the prefix associated with the segment. This path is typically a multi-hop path.

The Prefix Segment is a fundamental building block of SR because of its very interesting properties:

- it is global, all nodes in the SR domain know what to do with packets that have a Prefix Segment as Active Segment
- it is multi-hop, a single segment spans multiple hops in the path, reducing the number of segments needed for a path and allowing to exercise equal cost paths that span multiple hops
- it is ECMP-aware, it natively uses all equal cost paths which are widely prevalent in typical IP and MPLS networks

HIGHLIGHT

While the SR specifications allow attaching a Local Segment to an IGP prefix, the terms “IGP-Prefix Segment” and “Prefix-SID” are specifically used for Global Segments attached to a prefix. This is the key basis of most of the SR solutions and designs envisaged currently.

The operator assigns a domain-wide unique Prefix-SID to prefixes in the domain, typically the loopback prefix of each node in the domain. The domain-wide uniqueness is a property it has in common with IP addresses, which also have to be unique within the domain. Also similar to IP addresses: Prefix-SIDs very rarely change. Therefore, it is expected that the allocation of Prefix-SIDs is done according to a process that is similar to IP address allocation. Following a company’s policy, the operator can manually allocate the Prefix-SIDs or use a centralized management entity to do that.

Automated Prefix-SID allocation

"It is clear that in some limited context, the Prefix-SID's could be automatically derived from their bound IP addresses (e.g. if all the prefixes are /32 in the range 1.1.1/24 then the decimal value of the last 8 bits of the IP address could be used as index in the SRGB).

Unfortunately, a generic automated rule does not exist.

Even if I would be designing in a context where an automated rule could be defined, I would not consider such Prefix-SID allocation scheme. I would always prefer a Prefix-SID allocation mechanism similar to the IP address allocation of router loopbacks. The cost is not significant as the allocation process only occurs once and then never changes.

The Prefix-SID of a prefix is the corner stone on which all of the solution is built. This corner stone must be stable and reliable.

As networks evolve in the SDN era, automation and programmability are becoming key for their provisioning and management. Tools like Cisco Network Services Orchestrator (NSO)[1] facilitate a centralized and streamlined provisioning model which also makes tasks such as IP address allocations (and as an extension their Prefix-SIDs for SR) easier and in a manner that reduces errors."

— Clarence Filsfils

The Prefix Segment follows the *shortest path* to the prefix associated with the segment. More generally this should be "follows the path as computed by the applied algorithm". By default, the applied algorithm is Dijkstra's Shortest Path First (SPF) algorithm, using the IGP link metric as link cost. The outcome of that algorithm is the well-known IGP shortest path. By default the Prefix Segment follows this IGP shortest path. However, the SR architecture offers the instrumentation to deviate from that, by associating a specific *algorithm* with each Prefix-SID. The first algorithm that is specified, and also the default algorithm, is the well-known IGP shortest path algorithm. A second algorithm that has been defined is the "Strict Shortest Path First" algorithm. This algorithm is based on the SPF algorithm, but in addition, Strict SPF requires that the forwarding strictly follows the Shortest Path. When using the regular SPF algorithm, local policies may locally alter the forwarding decision as computed by the SPF algorithm. A well-known example where a local policy alters the forwarding decision is “autoroute announce” or “IGP shortcut”. Autoroute is a local functionality enabled on a TE tunnel which essentially tells the TE tunnel head-end to use the TE tunnel to reach the tail-end of the tunnel and destinations downstream of the tail-end. IGP installs the forwarding entries to these destinations via the TE tunnel instead of via their regular shortest path. When using the Strict SPF algorithm Prefix-SIDs, forwarding will strictly stay on the shortest path and will not be directed over the tunnel. Other algorithms can be defined in the future. This book assumes algorithm 0 (SPF) for a Prefix-SID unless specifically indicated differently.

To ensure the uniqueness of Prefix-SIDs, a couple of rules must be followed when associating a Prefix-SID with a prefix. Within the context of an IGP instance and a topology:

- a prefix can have only one associated Prefix-SID per algorithm type
- a Prefix-SID can only be associated with a single prefix

For example, a prefix 1.1.1.1/32 can have an associated prefix-SID 1001 for algorithm 0 and an associated prefix-SID 2001 for algorithm 1. Prefix 1.1.1.1/32 cannot have an associated Prefix-SID 1001 for both algorithm 0 and algorithm 1.

In another example, if Prefix-SID 1002 is associated with prefix 2001::1:1:1:2/128 for a specific algorithm, then Prefix-SID 1002 cannot be associated with another prefix for any algorithm.

HIGHLIGHT

The SR specifications allow assigning Prefix-SIDs to *any* prefix, not only host prefixes. In practice Prefix-SIDs are likely only assigned to host prefixes such as loopback address corresponding to a node's "router identifier" or other loopback addresses associated with certain service terminations (e.g. loopback used for BGP peering).

Each node in the SR Domain installs the forwarding entry for each Prefix Segment it receives. A node learns a Prefix-SID *S* advertised with an Algorithm *A* and associated with an IP prefix *P*. *N* is the nexthop towards prefix *P* along the path computed with the algorithm *A*. The node and its nexthop *N* both support algorithm *A*. If multiple equal cost paths exist towards prefix *P*, then multiple nexthops (*N1*, *N2*, …) exist and traffic is load-balanced over these equal cost paths. The node installs the following SR forwarding entry for this Prefix Segment:

- Incoming Active Segment: *S*
- Outgoing interface: interface towards nexthop *N*
- Nexthop: *N*
- Segment List Operation: If the nexthop *N* is the originator of *P* and *N* instructed to remove the Active Segment, then NEXT[2]. Else CONTINUE

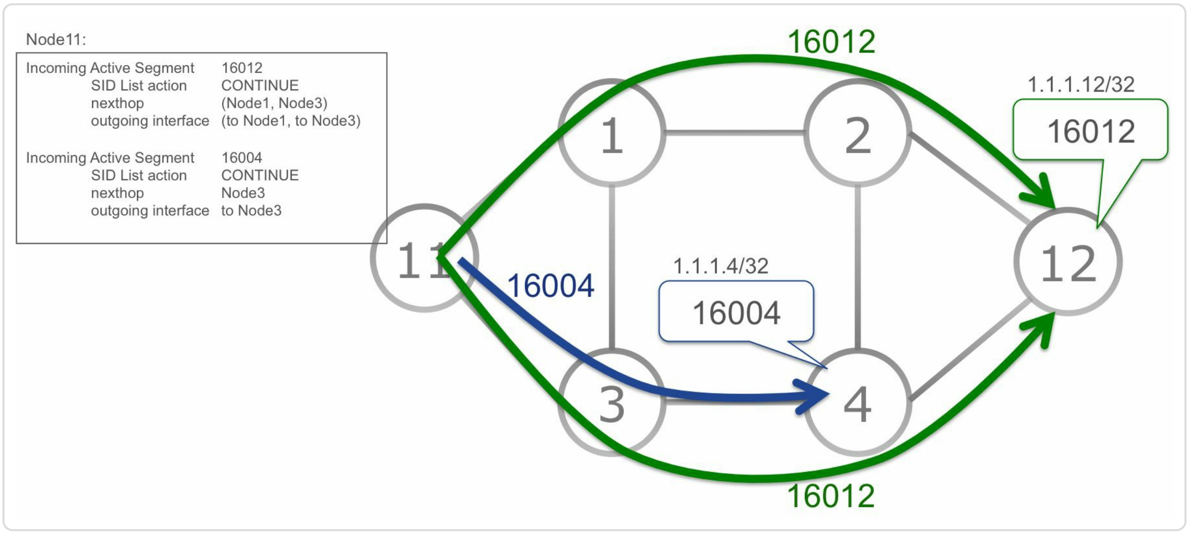
HIGHLIGHT

With a single Prefix-SID, one instructs the SR domain as a whole, to "forward across all the ECMP path along the shortest path to this prefix". A prefix-SID is a global segment.

The benefits of a Prefix-SID are:

- ECMP nature
- Automatically computed and maintained by the dynamic routing protocol (ISIS, OSPF, BGP). This is much better than Openflow, as an SR-based controller does not need to recompute and update all the related FIB entries across the SR domain upon any network topology change. A Software-Defined Networking (SDN) application relying on SR-enabled network only has to select the Prefix-SIDs to use, combine them and maintain a single per-flow state at the source or at the closest edge point to the source
- Automatically 50 msec protected. As we will see in the TILFA chapter, the dynamic routing protocol computes the primary and the backup paths of each Prefix-SID. Aside the SLA benefit (<50 msec loss of connectivity), an SR operator collects significant operational simplicity.

The network topology in Figure 2-1 is used to illustrate the Prefix-SID forwarding behavior.



```
Node11:
Incoming Active Segment    16012
         SID List action   CONTINUE
         nexthop           (Node1, Node3)
         outgoing interface (to Node1, to Node3)

Incoming Active Segment    16004
         SID List action   CONTINUE
         nexthop           Node3
         outgoing interface to Node3
```

Figure 2-1: Prefix-SID forwarding example

16012 is the Prefix-SID associated with the loopback prefix 1.1.1.12/32 of Node12 in this topology. A packet injected anywhere in the network with SID 16012 will reach Node12 via the ECMP-aware IGP shortest-path.

For example, Node11 steers a packet to Node12 using Node12's Prefix-SID 16012. Node11 has two equal cost paths to Node12: via Node1 and via Node3. Node11 load-balances the traffic over both paths using the regular load-balancing hash calculation. Node1 for example, receives the packet with Prefix-SID 16012 and steers it on the shortest path to Node12: via Node2. Node2 then forwards the packet to Node12.

16004 is the Prefix-SID associated with Node4's loopback prefix 1.1.1.4/32. Node11 steers a packet to Node4 using its Prefix-SID 16004. The shortest path to Node4 is via Node3. Node3 then forwards the packet to Node4.

2.3.1.1.1 IGP Node Segment

An IGP Node Segment (or commonly abbreviated to just "Node Segment" or "Node-SID") is a special type of IGP Prefix Segment. A Node Segment is special because it is associated with a prefix that identifies a specific node, typically a host prefix on a loopback interface of that node, which would typically be used as a "router identifier" for that node in IGP (e.g. OSPF router-id, MPLS router-id, etc.).

Since a Node-SID is a sub-type of a Prefix-SID, the instruction of a Node-SID is identical to that of the Prefix-SID: "*steer traffic along the ECMP-aware shortest path to the prefix associated with this segment*". There is no difference in forwarding behavior between a Prefix-SID and a Node-SID. The difference between a Node-SID and a Prefix-SID is in the control plane. Contrary to a Prefix-SID can a a Node-SID only be associated with a host prefix. A Node-SID is advertised with a special indication that it corresponds to a node (with the N-flag set) while other Prefix-SIDs are advertised without this indication (it has the N-flag unset).

The Prefix-SIDs 16004 and 16012 in Figure 2-1 can be Node-SIDs, if the associated loopback prefixes 1.1.1.4/32 and 1.1.1.12/32 identify respectively Node4 and Node12.

HIGHLIGHT

In the IGP context, in most cases, there will be a need for a single Node-SID to be setup on every router using loopback address corresponding to that router's "identifier". This would typically be the loopback interface host address that is used as the router-id for OSPF and also used by traditional MPLS signaling protocols as their router-id.

2.3.1.1.2 IGP Anycast Segment

Assigning a unicast IP address on more than one node in the domain makes that unicast prefix an anycast prefix. Anycast prefixes are syntactically indistinguishable from unicast IP prefixes. An IGP Anycast Segment (or commonly simplified to just "Anycast Segment", or "Anycast-SID") is a special type of IGP Prefix Segment that is associated with an anycast prefix. Such anycast prefix, with its associated Prefix-SID, is not identifying a specific node, rather a group of nodes. Hence such Anycast-SID is not a Node-SID and must be originated with the N-flag unset. Thus a Prefix-SID even if attached to a host prefix and typically configured on a loopback address, is not always identifying a node.

An anycast set or anycast group is the set of two or more nodes that originate the same (anycast) address. The property of an anycast set is that traffic destined to the anycast address of an anycast set is routed to the closest (shortest IGP distance) node of the set. If two or more members of the anycast set are located at the same distance, then the traffic is load-balanced between these equal distance members. This mechanism distributes the traffic load over all members of the anycast set.

Anycast sets also provide a protection mechanism against failure of one or more members of the anycast set. The remaining members in the set seamlessly take over and the traffic source and the other nodes in the network steer packets destined to the anycast prefix via the new shortest path to the member of the anycast set that is now closest.

Anycast addresses are commonly known for their usage with connectionless services, such as Domain Name System (DNS) servers. Using anycast addresses provides load-sharing and redundancy for these services.

Projecting this onto SR, an Anycast-SID is associated with an anycast prefix. Since an Anycast Segment is a sub-type of a Prefix Segment, the instruction of an Anycast Segment is identical to that of the Prefix Segment: "*steer traffic along the ECMP-aware shortest path to the prefix associated with this segment*". For an Anycast Segment this comes down to steering the traffic on the ECMP-aware shortest-path towards the closest node of the anycast set.

An Anycast Segment inherits the properties of the underlying anycast prefix: load-sharing and redundancy. Traffic to an Anycast Segment is steered to the closest originator of the Anycast-SID. If two or more nodes of the anycast set are at the same distance then the traffic is load-balanced between these equally distant members. If a member of the anycast set fails, then the remaining members of the set handle traffic to the Anycast Segment.

Anycast Segments are also useful to express macro-engineering policies such as: *steer traffic via the anycast set "plane A"* or *steer traffic via the anycast set "transit nodes group A"* or *steer traffic via the anycast set "Central Europe"*. These policies do not indicate a specific transit node, but a group of transit nodes, any of the nodes in the group fulfills the traffic steering requirement. These groups of transit nodes are made members of anycast sets and are targeted by their Anycast-SID.

The network topology in Figure 2-2 illustrates the forwarding behavior of an Anycast Segment.

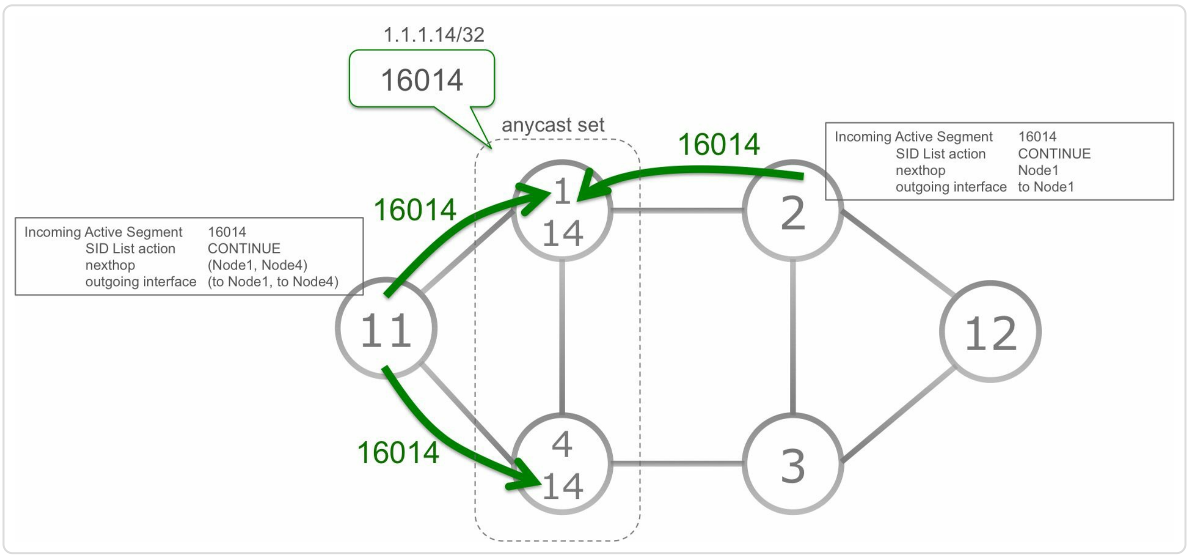


Figure 2-2: Forwarding characteristics of an Anycast Segment

16014 is the Anycast-SID associated with the anycast prefix 1.1.1.14/32 configured on a loopback interface of both Node1 and Node4 in this topology. By configuring the same anycast prefix and Anycast-SID on both nodes, they are member of the anycast set with Anycast-SID 16014. A packet injected anywhere in the network with Segment ID 16014 will reach Node1 or Node4, whichever is the closest, via the ECMP-aware IGP shortest-path.

In Figure 2-2, Node2 receives a packet destined to Anycast-SID 16014. Node2’s shortest path to 1.1.1.14/32 is towards Node1. Node2 forwards all packets sent to the Anycast-SID 16014 towards Node1.

In the same Figure 2-2, Node11 receives a packet destined to Anycast-SID 16014. Node11 sees prefix 1.1.1.14/32 as reachable via both Node1 and Node4, both are located at the same distance of Node11. Node11 installs both paths in the forwarding table and load-balances traffic to Anycast-SID 16014 over both paths using the regular load-balancing hash calculation.

If Node1 fails, as illustrated in Figure 2-3, the network converges. Node2 and Node11 now see Node4 to be the new closest node (and only one remaining in this example) that originates 1.1.1.14/32 and its associated Prefix-SID 16014. Node2 and Node11 now forward packets destined to Anycast-SID 16014 towards Node4. This happens automatically without requiring any intervention on any node.

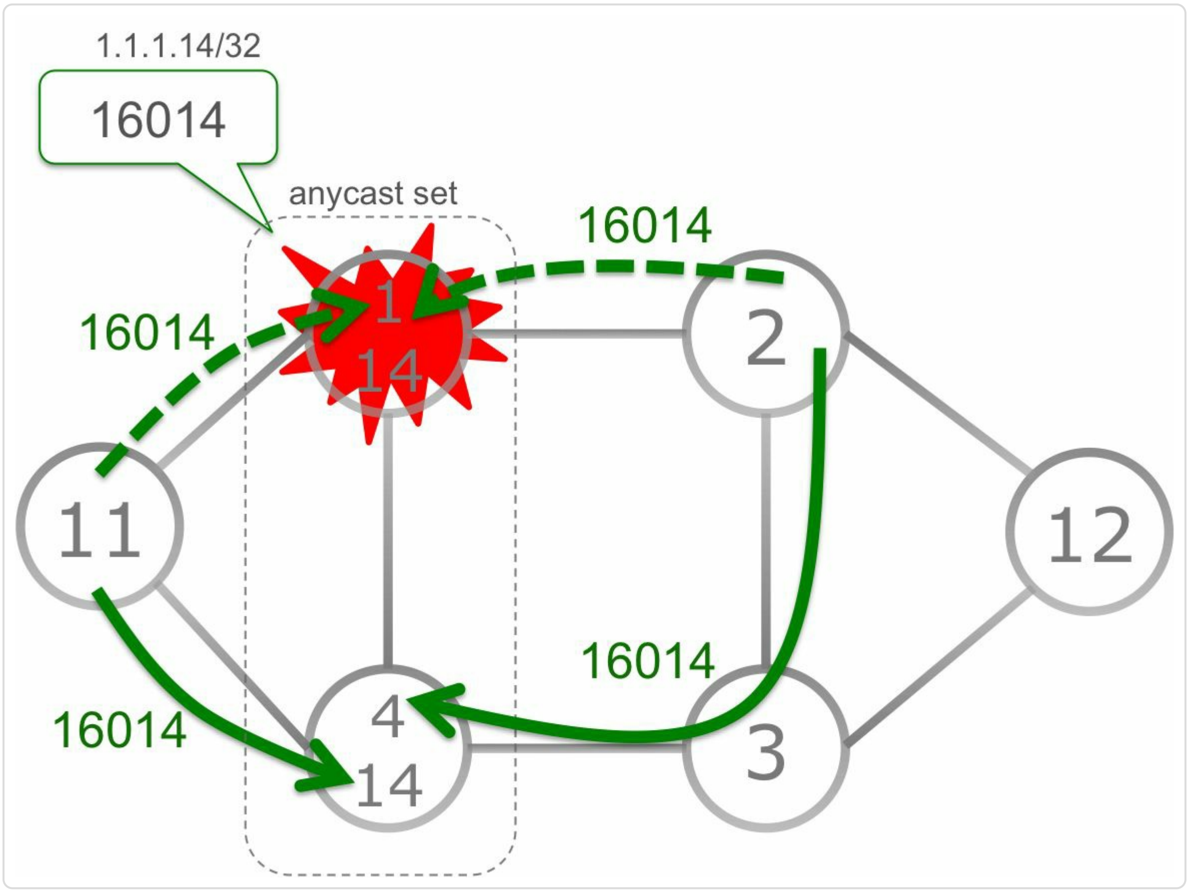


Figure 2-3: Anycast-SID redundancy functionality

Since there is a one-to-one mapping between prefix and Prefix-SID,all nodes in the anycast set must advertise the same Anycast-SID with the anycast prefix.

When using Anycast-SID with the MPLS data plane implementation of SR, there are certain special considerations that need to be taken care of and this is described in chapter 4, “Management of the SRGB”.

HIGHLIGHT

An Anycast-SID is a Prefix-SID associated to an Anycast IP address. With a single Anycast-SID, one instructs the SR Domain, as a whole, to “forward across all the ECMP path along the shortest path to the closest instance of the anycast prefix”. An Anycast-SID is a global segment.

The benefits of an Anycast-SID are:

- ECMP nature
- High-Availability (HA): when the closest instance of an anycast address gets disconnected from the network, the SR Domain automatically forwards a packet bound to this Anycast-SID to the next closest instance of its related IP anycast address.
- Traffic Engineering based on "Macro expression". Often, an operator does not need (or want) to traffic-engineer a demand on a unique path through the network. Often, the operator only wants to get the demand routed via a given region (e.g. Germany) or via a given router function in a given region (e.g. "via any backbone router in the Netherlands", or "via all the Spines of this Data Center"). Anycast-SIDs can be allocated to all the routers in a region or to all the routers of a region that perform a given function. Anycast-SIDs enable these "macro TE expressions" and deliver benefits in terms of High Availability (HA) and ECMP.

2.3.1.2 IGP Adjacency Segment

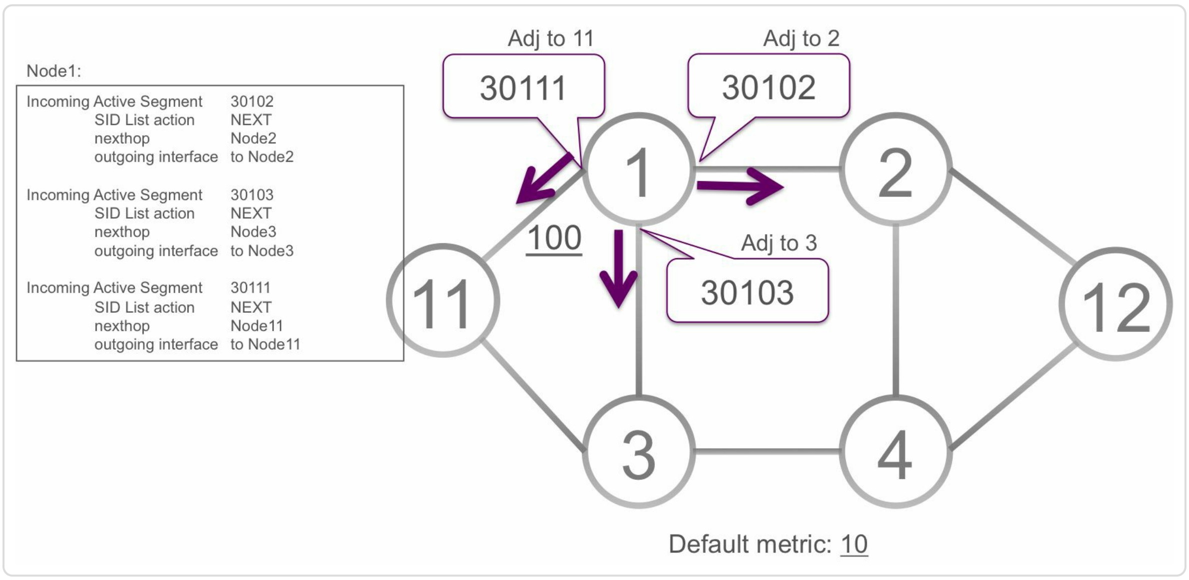
An IGP Adjacency Segment (or commonly abbreviated as "Adjacency Segment", or "Adjacency-SID", or even "Adj-SID") is a segment that is associated with a unidirectional adjacency or set of unidirectional adjacencies. The instruction of an IGP Adjacency Segment is "*steer traffic out of the link(s) of the adjacency(ies) associated with this segment*". The traffic on this Adjacency Segment is steered on the link regardless of shortest path routing. An Adjacency Segment is typically used to steer traffic to a neighbor node if the IGP shortest path to that neighbor is not via that direct link, otherwise the Prefix Segment of that neighbor is generally preferred.

Typically an Adjacency Segment is a Local Segment, local to the node that advertises it. This book uses the term "IGP Adjacency Segment" to indicate a Local IGP Adjacency Segment, otherwise it is preceded by the term “Global”.

HIGHLIGHT

While the SR specifications allow an IGP Adjacency Segment to be a Global Segment, typically these segments are used as Local Segments so as to reduce the amount of forwarding state that would need to be programmed on devices across the network. Note that all Global Segments need to be programmed in the forwarding plane of every router and doing so for Adjacency Segments can increase the state significantly without giving much added benefits.

The network topology in Figure 2-4 illustrates the forwarding behavior of an Adjacency Segment.


Node1:

```
Incoming Active Segment   30102
          SID List action NEXT
          nexthop         Node2
          outgoing interface to Node2

Incoming Active Segment   30103
          SID List action NEXT
          nexthop         Node3
          outgoing interface to Node3

Incoming Active Segment   30111
          SID List action NEXT
          nexthop         Node11
          outgoing interface to Node11
```

Figure 2-4: Adjacency Segment forwarding

Node1 allocated 30102 as the Adjacency Segment ID associated with the adjacency of Node1 towards Node2. And Node1 allocated 30103 as the Adjacency Segment ID associated with the adjacency of Node1 towards

Node3. Node1 steers packets that it receives with Active Segment 30102 on the link of the adjacency to Node2. It steers packets with Active Segment 30103 on the link of the adjacency to Node3. The Adjacency-SIDs are local to Node1; only Node1 correctly understands them. Other nodes can use the same Adjacency-SID values, but their meaning will be different. Each node will locally define their meaning.

The link of Node1 to Node11 has an IGP link metric 100. Hence the shortest path from Node1 to Node11 does not go via that direct link, but via Node3. The Adjacency Segment of the link from Node1 to Node11 is needed to steer packets on that link.

To make use of local Adjacency-SIDs, they are typically preceded in the SID list by the Node's Prefix-SID. A packet injected on any node in the topology of Figure 2-4 with a SID list {16001, 30103} will be forwarded along the shortest path to Node1 and then switched on the link towards Node3.

If Node1 would advertise a Global Adjacency Segment (e.g. 20103) for its adjacency to Node3, then the above SID List could be reduced to {20103} with the same result. However, the reduction of the size of the SID List comes at the cost of increased state creation on all nodes in the network. If the Adjacency Segment is a Global Segment then all nodes in the network need to install the forwarding entry for it.

A node allocated an Adjacency-SID *S* associated with its IGP adjacency on interface *I*. The node installs the following entry in the forwarding table:

- Incoming Active Segment: *S*

- Segment List Operation: NEXT
- Outgoing interface: interface *I*

There can be more than one Adjacency-SID assigned to a given link such that each of these Adjacency-SID have some different properties associated with it. We shall see one of these applications further in chapter 9, “Topology Independent LFA (TI-LFA)”.

HIGHLIGHT

An Adjacency-SID is a SID associated with a local interface of a specific router. Upon receiving a packet with a local Adj-SID as active segment, the router forwards the packet out of the related interface after having activated the next SID in the SID list. An Adj-SID is not understood by the SR Domain as a whole but it is only recognized by the router which originates it. An Adj-SID is a local segment.

The benefits of an Adjacency-SID are:

- Complete explicit routing solution; *any* path can be expressed by a list of segments.
- Time-Division Multiplexing (TDM) migration: Adj-SID contributes to the migration of services that were previously relying on TDM transport to SR transport. Adj-SIDs do not rely on IP multi-hop dynamic routing. Adj-SIDs allow picking a specific non-ECMP path out of a set of ECMP paths. A list of Adj-SIDs expresses an explicit path without ECMP and without relying on dynamic IGP routing. These properties are often requested by services that used to be transported by TDM transport. While most of the traffic transported over an SR domain would be IP-based and would use one or few prefix-SIDs, a small percentage of the traffic could be transported “à la TDM” using a longer list of Adj-SIDs.

“We try to rarely use Adj-SIDs. They do not support ECMP and they only express one single link hop. We prefer expressing an explicit path with Prefix-SIDs. We get shorter SID lists and we spread the demand over all the ECMP paths of the Prefix-SIDs."

— Clarence Filsfils

2.3.1.2.1 Layer-2 Adjacency-SID

Link bundles or Link Aggregation Groups (LAG) are widely used. A LAG combines multiple Layer-2 interfaces in a single Layer-3 interface. Since an IGP adjacency is only established over the Layer-3 interface, it reduces the number of IGP adjacencies that need to be maintained when multiple parallel links between neighbors are used. Traffic over the LAG is load-balanced over all its member links. If there is a requirement to steer traffic over specific individual physical links, then a Layer-2 Adjacency-SID is allocated for each bundle member. This Layer-2 Adjacency-SID is then used to steer packets over individual members, equivalent with (Layer-3) Adjacency-SIDs.

The network topology in Figure 2-5 illustrates the forwarding behavior of a Layer-2 Adjacency Segment. The Layer-3 link between Node1 and Node3 consists of a Link Bundle with three links.

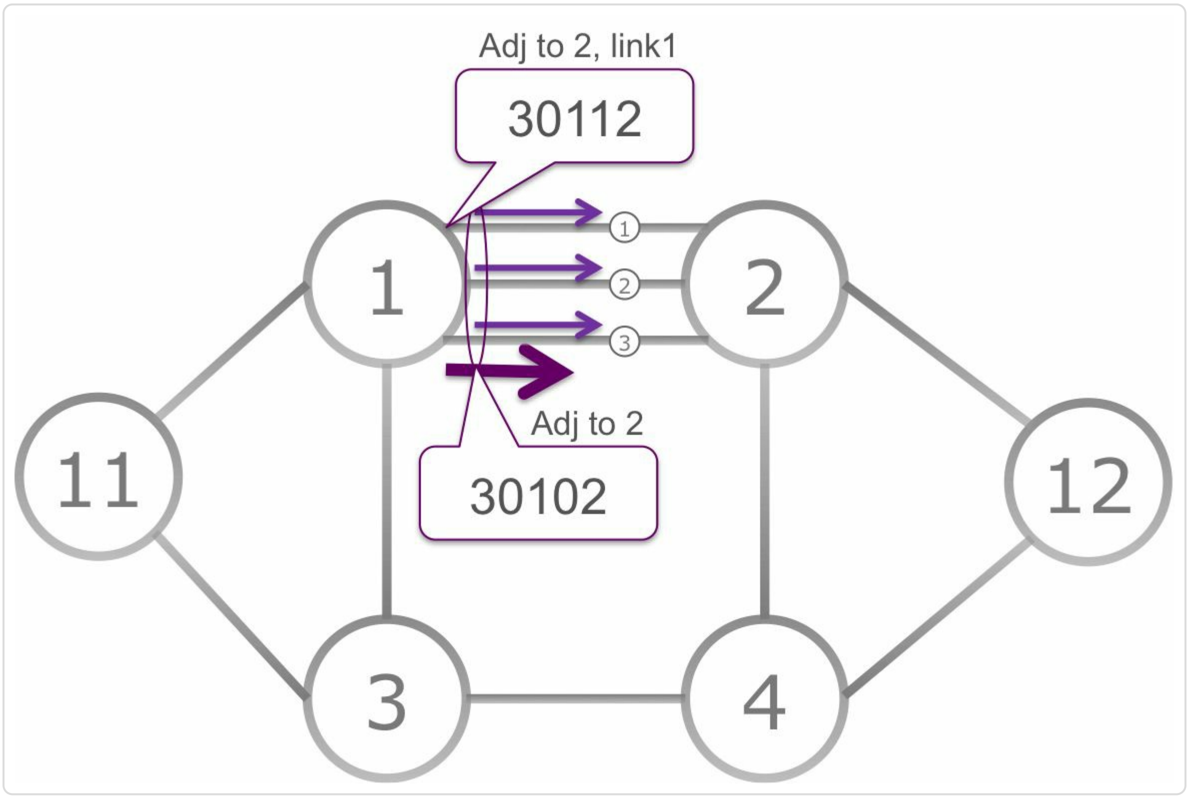


Figure 2-5: Layer-2 Adjacency-SID

Node1 allocated 30102 as the (Layer-3) Adjacency-SID for the adjacency of Node1 towards Node2. Node1 also allocated Layer-2 Adjacency-SIDs for each individual bundle member link. For example, Node1 allocated L2 Adjacency-SID 30112 for link 1 in the bundle. Node1 steers packets that it receives with Active Segment 30112 on link 1 of the bundle to Node2. Node1 load-balances packets with Active Segment 30102 over all links of the bundle to Node2.

HIGHLIGHT

An L2 Adjacency-SID is an Adj-SID that is associated with a specific member link of an L2 bundle.

2.3.1.2.2 Group Adjacency-SID

In some designs, there are multiple parallel links between a pair of routers and these are kept as separate links rather than bundling them together into a LAG link for the IGPs. In such cases, it is sometimes desirous to group all the individual adjacencies to the neighboring router and represent them with a single Group Adjacency-SID which is then an instruction to send traffic in a load-balanced manner over the constituent links.

This type of Adjacency-SID has not yet come into focus in Segment Routing use-cases as of writing of this book and is not covered here if further detail.

2.3.2 BGP Segments

BGP can also be used as the SR control plane to distribute the SIDs in the network. The details of the SR BGP implementation are explained in chapter 7, “Segment Routing BGP Control Plane”.

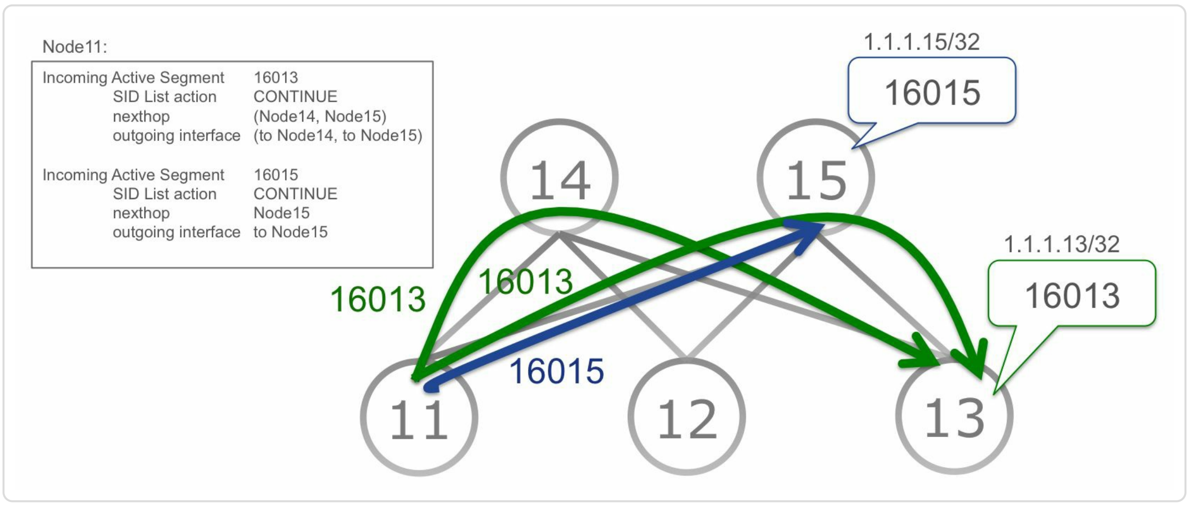
Some networks, notably Massive Scale Data Centers (MSDC), use only BGP as the routing protocol. In a nutshell, the argumentation[3] is that it is simple, allows for per-hop traffic engineering and is supported by practically all vendors (“BGP is the Better IGP”). But the applicability of BGP Segment Routing is not restricted to those networks and signaling of labeled unicast prefixes via BGP (using RFC3107 mechanisms) is prevalent in many Service Provider networks

2.3.2.1 BGP Prefix Segment

A BGP Prefix Segment is associated with a BGP prefix, similar to the IGP Prefix Segment that is associated with an IGP prefix. Equivalent to the IGP Prefix Segment is a BGP Prefix Segment a Global Segment, understood by all nodes within the SR BGP Domain. The instruction of a

BGP Prefix Segment is "*steer traffic along the ECMP-aware BGP multi-path to the prefix associated with this segment*". The BGP Prefix Segment load-balances traffic over the available BGP multi-paths.

Figure 2-6 illustrates the BGP Prefix-SID forwarding behavior in a small three-stage (two levels) Data Center topology.



```
Node11:
Incoming Active Segment    16013
          SID List action  CONTINUE
          nexthop          (Node14, Node15)
          outgoing interface (to Node14, to Node15)

Incoming Active Segment    16015
          SID List action  CONTINUE
          nexthop          Node15
          outgoing interface to Node15
```

Figure 2-6: BGP Prefix-SID forwarding example

16013 is the BGP Prefix-SID associated with the loopback prefix 1.1.1.13/32 of Node13. A packet injected anywhere in the network with Segment ID 16013 will reach Node13 via the ECMP-aware BGP best-path.

For example, Node11 receives packets with Node13's BGP Prefix-SID 16013 as Active Segment. Node11 has two paths in the BGP multi-path to Node13's loopback prefix: via Node14 and via Node15. Node11 load-balances the traffic over both paths using the regular load-balancing functionality. Node14 receives the packet with BGP Prefix-SID 16013 and steers it on the BGP best path to Node13.

16015 is the BGP Prefix-SID associated with Node15's loopback prefix 1.1.1.15/32. Node11 receives a packet with Node5's BGP Prefix-SID 16005 as Active Segment. The BGP best path to Node5's loopback prefix is to Node15. Node11 forwards the packet to Node15.

2.3.2.1.1 BGP Anycast Segment

Anycast prefixes also exist in BGP. Different nodes can advertise the same BGP prefix, this way making such prefix an anycast prefix. Same as for the IGP Anycast Segment, the BGP Anycast Segment provides coarse-grained traffic steering capabilities such as "*steer traffic via spine nodes in group A*". It also has the redundancy property where the other members in the anycast set cover failure of a member of the anycast set.

Figure 2-7 illustrates the BGP Anycast-SID forwarding behavior in a small 2-stage Data Center topology.

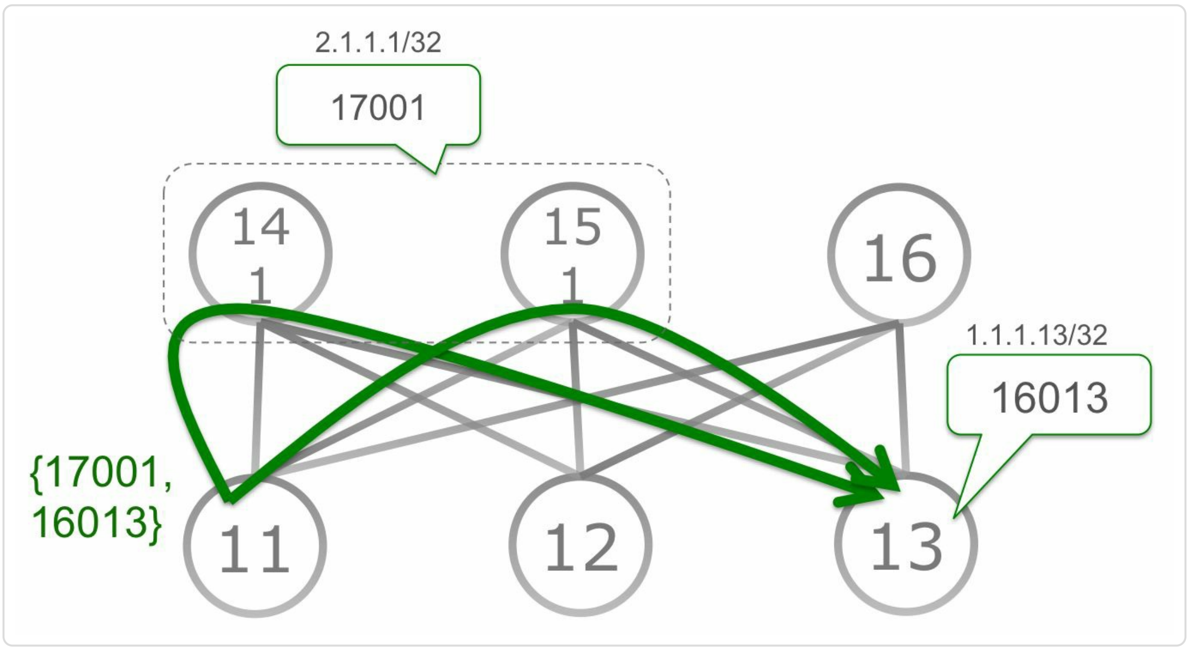


Figure 2-7: BGP Anycast-SID forwarding example

Node14 and Node15 are members of an anycast set; they both advertise anycast prefix 2.1.1.1/32 with BGP Anycast-SID 17001. Both nodes also advertise their own BGP Prefix-SID, 16014 and 16015 for Node14 and Node15 respectively. Node11 wants to send traffic to Node13, but excluding spine Node16. Therefore Node11 adds two Prefix-SIDs in the packet Segment List: {17001, 16013}. This SID List first steers the packets to any node of the anycast set, which is any of the nodes advertising BGP Anycast-SID 17001, and then to the destination Node13 using Node13's BGP Prefix-SID. Traffic to the BGP Anycast-SID will be load-balanced over the BGP multi-path with one path via Node14 and another path via Node15. Upon failure of e.g. Node14, all traffic to BGP Anycast-SID 17001 will go towards Node15.

2.3.2.2 BGP Peer Segments

BGP Peer Segments are associated with BGP peering sessions to a specific neighbor or a set of neighbors. They are typically Local Segments which are assigned by a BGP speaker to its peering sessions and signaled via BGP Link-State (BGP-LS) address family (RFC7752) in BGP itself. BGP Peer Segments are applicable for both internal as well as external BGP sessions.

We will get into details of BGP Link-State extensions and these BGP Peer Segments in the next part of this book which will focus on SR Traffic Engineering where these Segments are used. Here, we will only provide a brief and high-level introduction of these Segments.

HIGHLIGHT

While the SR and BGP-LS specifications allow attaching a BGP Peering Segment to be a Global Segment, typically these segments are used as Local Segments. This is because using Global Segment would introduce the challenge of propagating this information to all routers (including ones that don't run BGP) and typically in many networks it would be the IGP Segments which deliver a Segment Routed packet to the BGP speaker who can then direct them towards the Peer(s) as indicated by the BGP Peer Segment.

2.3.2.2.1 BGP Peer Node Segment

A BGP Peer Node Segment (or BGP Peer-Node-SID) is associated with a peering session to its neighbor. The instruction of a BGP Peer Node Segment is "*steer traffic to the specific BGP peer node via ECMP multi-path towards that peer router*". The BGP Peer Node Segment overrules the traditional BGP decision process and ensures that the traffic is sent to the specific peer BGP node over available underlying (IGP) multi-paths.

Figure 2-8 illustrates the BGP Peer-Node-SID in a typical service provider BGP peering setup.

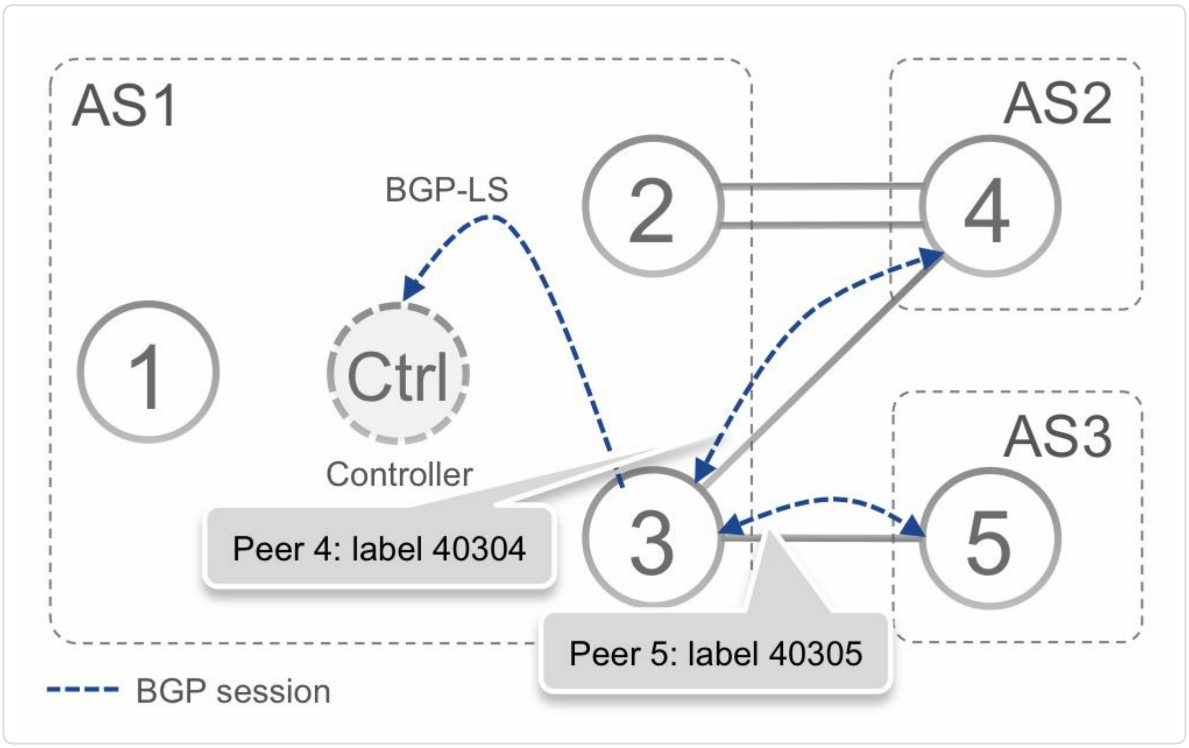


Figure 2-8: BGP Peer Node SID example

On Node3, BGP peers with Node4 in AS2 and Node5 in AS3. It allocates Local Segment with label 40304 and 40305 respectively for its peers Node4 and Node5 and advertises these labels via BGP-LS. Any Segment Routed packet that arrives on Node3 with the label 40304 will be forwarded out towards Node4 in AS2 overruling the BGP decision process.

2.3.2.2.2 BGP Peer Adjacency Segment

A BGP Peer Adjacency Segment (or BGP Peer-Adj-SID) is associated with a specific underlying path or link towards a specific neighbor node. It is likely that there are multiple links connecting two BGP routers that have an External BGP session between them. The instruction of a BGP Peer Adjacency Segment is "*steer traffic to the specific BGP peer node via the specified interface towards that peer router*". The BGP Peer Adjacency Segment overrules the traditional BGP decision process and also perhaps the ECMP or other local routing decision that picks the actual link over which to send the packet to the BGP neighbor.

Figure 2-9 illustrates the BGP Peer-Adj-SID in a typical service provider BGP peering setup.

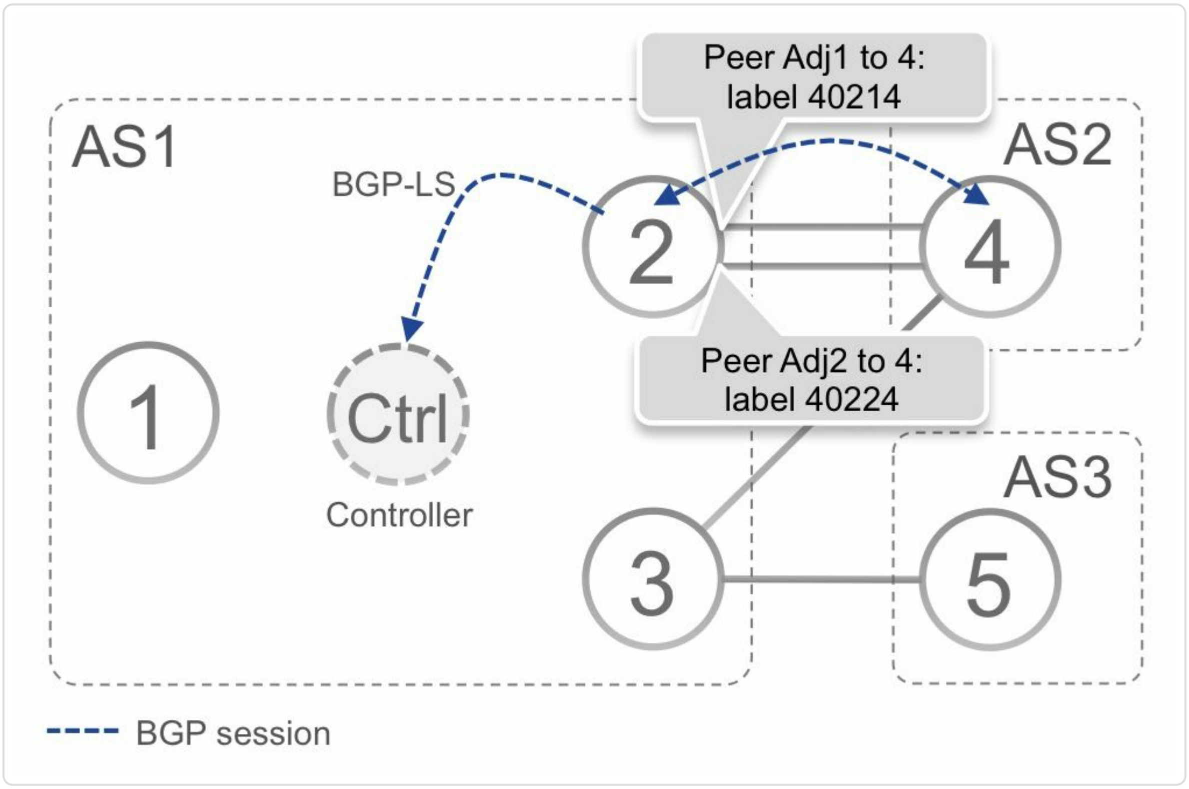


Figure 2-9: BGP Peer Adjacency SID example

On Node2, BGP peers with Node4 in AS2 (a multi-hop EBGP session between their loopbacks) and there are two underlying links connecting Node2 to Node4 – link 1 and link 2. Node2 allocates Local Segment with label 40214 and 40224 respectively for its peering with Node4 for link 1 and 2. Any Segment Routed packet that arrives on Node2 with the label 40214 will be forwarded out towards Node4 in AS2 only over link 1.

2.3.2.2.3 BGP Peer Set Segment

A BGP Peer Set Segment (or BGP Peer-Set-SID) is associated with a group of neighbors with which peering sessions exist. The instruction of a BGP Peer Set Segment is "*steer traffic to the specific BGP peer node(s) that are members of the peer group via ECMP BGP multi-path*". The BGP Peer Set Segment overrules the traditional BGP decision process and ensures that the traffic is sent to the specific peers' BGP node(s) using BGP multi-path mechanisms.

Figure 2-10 illustrates the BGP Peer-Set-SID in a typical service provider BGP peering setup.

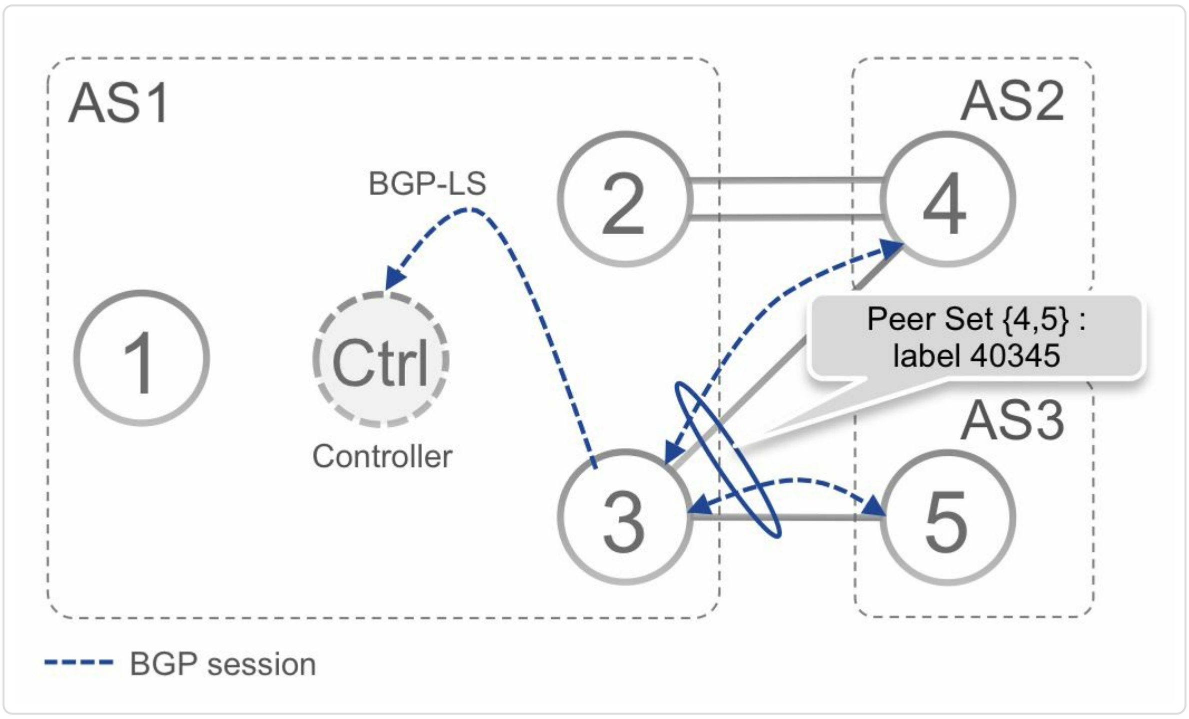


Figure 2-10: BGP Peer Set SID example

On Node3, BGP peers with Node4 in AS2 and Node5 in AS3. It allocates Local Segment with label 40345 to represent the peer set including Node4 and Node5 and advertises it via BGP-LS. Any Segment Routed packet that arrives on Node3 with the label 40345 would get forwarded out towards Node4 and Node5 using BGP multi-path ECMP mechanism overruling the BGP decision process.

2.3.3 Combining Segments

Segments are the building blocks in Segment Routing. These building blocks can be combined to steer packet flows via any end-to-end path through the network. Different types of Segments can be combined in such end-to-end path, such as different types of IGP Segments and BGP

Segments. Since a segment can be any kind of instruction, topological or service-based, the path can for example also steer the packet through a chain of services.

TILFA FRR and TE applications such as disjointness, latency or bandwidth optimization require explicit paths. The real-live data set analysis confirms that shallow SID lists (few SIDs) are required to express these policies. Please refer to the TILFA and TE sections for the detailed analysis or the related publicly available documents ([draft-francois-rtgwg-segment-routing-ti-lfa] and [SIGCOMM]).

HIGHLIGHT

Most often, a SID list is made of one or two SIDs. We rarely need more than 3 or 4 SID's to express the TILFA/TE policies in real networks. This is confirmed by the analysis of real-live data sets.

"Network topologies often reflect the same properties as rail-way/highway networks. It was a fundamental intuition at the origin of the SR project that few SIDs would be required to express the real policies of real networks. This intuition came when thinking about the best path to Rome from Brussels (shortest path versus a detour via Geneva to avoid the traffic jam in the Gottardo tunnel). Human beings are able to express their transport policies in terms of a small number of shortest paths via intermediate cities (instead of a very large number of turn-by-turn directions). The same should be possible for applications in an SR-enabled network."

— Clarence Filsfils

The topology in Figure 2-11 illustrates how segments can be combined to define an end-to-end path from a Data Center over the WAN.

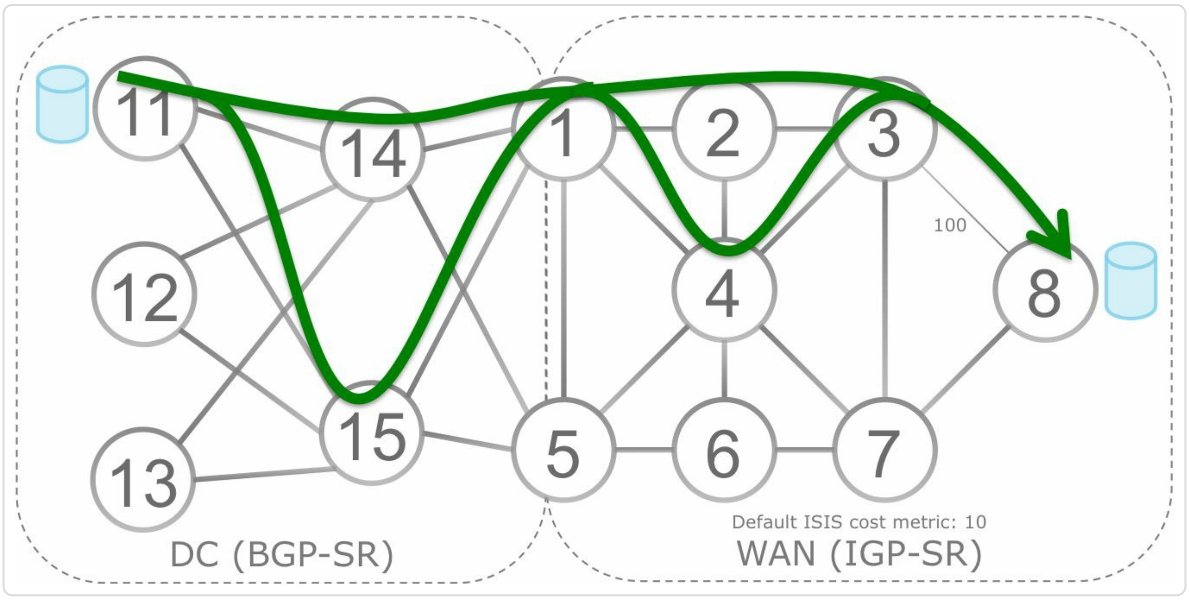


Figure 2-11: Combine segments to define end-to-end path

Applications on a number of servers in the Data Center need to communicate with applications on another set of servers in a remote location, beyond the WAN. The Data Center servers are located next to Node11 and the remote servers are located next to Node8. The traffic flows between the servers has to follow certain requirements that are not detailed in this example. Based on these traffic requirements, a path is calculated. The path that fulfills the requirements of the traffic flows, passes through border node Node1, on the border between Data Center and WAN. The path then passes Node3 in the WAN and then follows a low latency and high metric link from Node3 to Node8. This end-to-end path can be expressed with a Segment List consisting of the following segments: a BGP Prefix Segment (16001) to Node1 in the Data Center, followed by an IGP Prefix Segment (16003) to Node3 in the WAN and finally, also in the WAN, an IGP Adjacency Segment (30308) from Node3 to Node8. The BGP Prefix Segment steers the flows via the ECMP-aware BGP best-path (BGP multi-path). The traffic flows use all available ECMP in the Data Center from Node11 towards Node1. From Node1 the flows follow the ECMP-aware IGP shortest path to Node3, following the IGP Prefix Segment of Node3. Also in the WAN all possible ECMP is used to go from Node1 to Node3. Finally the traffic flows are steered over a path from Node3 to Node8 that is not the IGP shortest path but traverses a high metric link. An IGP Adjacency Segment is used to accomplish this. This IGP Adjacency Segment steers the traffic over a link regardless of the IGP shortest path.

Another set of applications has different requirements and the traffic flows for these applications will follow a different path. Note that even though a different path is used, the amount of state in the network stays the same. This is true regardless how the traffic flows are steered in the network and regardless of the amount of traffic flows through the network.

Figure 2-12 illustrates this other end-to-end path.

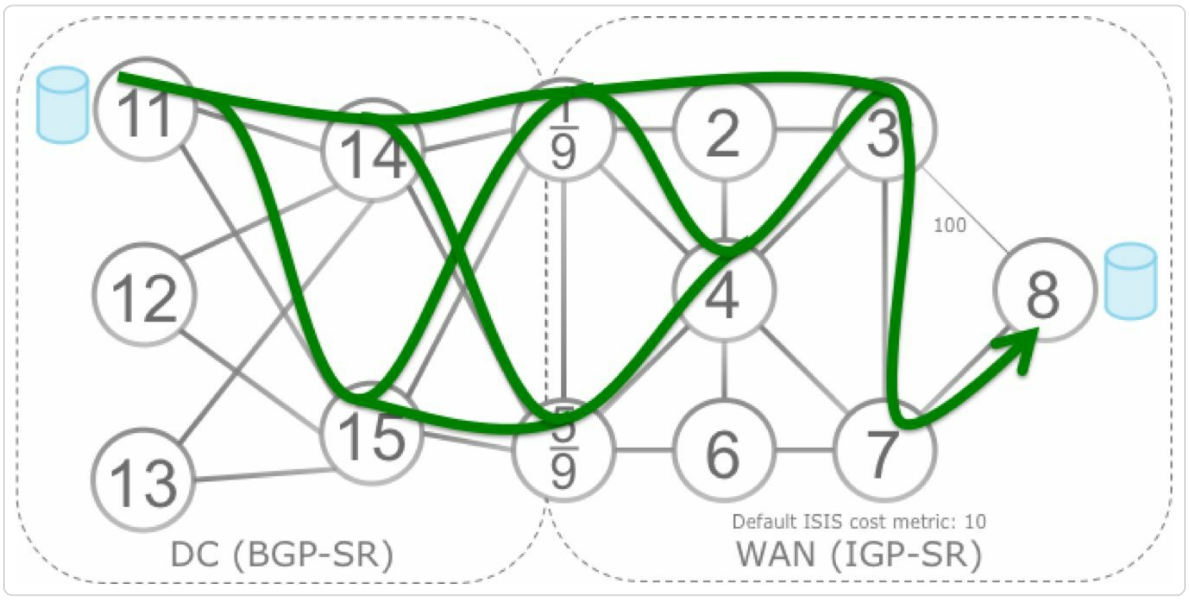


Figure 2-12: Combine segments to define another end-to-end path

This path uses more of the available ECMP in the Data Center by first steering the traffic flows to the BGP Anycast Segment (17009) of the border nodes anycast set. The traffic flows are load-balanced over both border nodes (Node1 and Node5) because of the BGP Anycast Segment. Within the Data Center all ECMP towards both border nodes is used for the traffic flows. Then the traffic flows pass through Node3 via the IGP Prefix Segment (16003) of Node3. The traffic flows use all available ECMP in the WAN from the border nodes to Node3. Then finally the traffic flows follow the IGP Prefix Segment to Node8.

Depending on the requirements of a service, its traffic flow can be steered on many other possible paths through the network by simply combining different segments (the Segment Routing building blocks) to define the end-to-end path. How the traffic flows are steered or the amount of traffic flows that are steered does not impact the forwarding state on the midpoint nodes in the network. The amount of state on the midpoint nodes stays constant. To steer the traffic on its path, the nodes in the network simply execute the instructions (the segments) that are encoded in the packet header.

2.4 Segment Routing Data Plane

The SR architecture does not assume a specific data plane implementation. The SR architecture can be instantiated on the MPLS data plane and on the IPv6 data plane.

SR on the MPLS data plane leverages the existing MPLS architecture (IETF RFC 3031). A Segment Identifier (SID) is represented as a MPLS label or an index value in a MPLS label space. The Segment List is represented in the packet header as a stack of MPLS labels. The SR MPLS data plane implementation can be used with both IPv4 and IPv6 address families. Both IPv4 and IPv6 control planes can program the MPLS forwarding entries.

SR can also be applied to the IPv6 data plane using a new type of IPv6 Extension Header: the Segment Routing Header (SRH). Extensions headers are defined in the base IPv6 standard specification (IETF RFC 2460) as an expansion mechanism to carry additional routing information. The IPv6 data plane implementation of Segment Routing is commonly known as “SRv6”. In the SRv6 implementation, a segment is represented as an IPv6 address. And a Segment List is encoded as an ordered list of IPv6 addresses in the SRH. SRv6 offers the SR functionality without requiring MPLS on the network.

2.5 Summary

- A segment is a topological or service (or some other) instruction.
- A Segment List is an ordered list of instructions.
- Source routing: the source, or an edge as close as possible to the source, encodes a list of segments in the header of a packet to steer the packet along the desired multi-area multi-domain topological and service path.
- The nodes in a network process a packet based on the instructions in its header.
- Per-flow state is only maintained at the source, not in the network.
- Prefix Segment: steers a packet along ECMP-aware shortest path to the segment's prefix. Global segment.
- Node Segment: Prefix Segment of a prefix that represents a node (i.e. router-id).
- Anycast Segment: steers a packet along ECMP-aware shortest path to the segment's anycast prefix
- Adjacency Segment: steers a packet on the link of the segment's adjacency. Typically a local segment.
- BGP Peer Segment: steers a packet towards a specific BGP peer (over a specific link) overruling BGP decision process. Typically a local segment.
- Segment Routing can be implemented on different data planes (MPLS and IPv6) and control plane protocols (ISIS, OSPF, BGP).

2.6 References

- [draft-francois-rtgwg-segment-routing-ti-lfa] Francois, P., Filsfils, C., Bashandy, A., and B. Decraene, "Topology Independent Fast Reroute using Segment Routing", draft-francois-rtgwg-segment-routing-ti-lfa (work in progress), May 2016, <https://datatracker.ietf.org/doc/ draft-francois-rtgwg-segment-routing-ti-lfa>.
- [draft-ietf-6man-segment-routing-header] Previdi, S., Filsfils, C., Field, B., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., and D. Lebrun, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header (work in progress), September 2016, https://datatracker.ietf.org/doc/draft-ietf-6man-segment-routing-header.
- [draft-ietf-isis-l2bundles] Ginsberg, L., Marcon, J., Filsfils, C., Previdi, S., Nanduri, M., Aries, E., "Advertising L2 Bundle Member Link Attributes in IS-IS" (work in progress), August 2016, <https://datatracker.ietf.org/doc/draft-ietf-isis-l2bundles>.- [draft-ietf-isis-segment-routing-extensions] Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions (work in progress), June 2016, https://datatracker.ietf.org/doc/draft-ietf-isis-segment-routing-extensions.
- [draft-ietf-ospf-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions (work in progress), July 2016,

<https://datatracker.ietf.org/doc/draft-ietf-ospf-segment-routing-extensions.

- [RFC2460] Deering, S. and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460, DOI 10.17487/RFC2460, December 1998, https://datatracker.ietf.org/doc//rfc2460.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, “Multiprotocol Label Switching Architecture”, RFC 3031, DOI 10.17487/RFC3031, January 2001, https://datatracker.ietf.org/doc/rfc3031.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, “North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP”, RFC 7752, DOI 10.17487/RFC7752, March 2016, https://datatracker.ietf.org/doc/rfc7752.
- [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://datatracker.ietf.org/doc/rfc7855>.- [draft-ietf-spring-segment-routing] Filsfils, C., Ed., Previdi, S., Ed., Decraene, B.,Litkowski, S., and R. Shakir, “Segment RoutingArchitecture”, Work in Progress, draft-ietf-spring-segment-routing-09, July 2016, <https://datatracker.ietf.org/doc/draft-ietf-spring-segment-routing>.- [draft-ietf-idr-bgp-prefix-sid] Previdi, S., Filsfils, C., Lindem, A., Patel, K., Sreekantiah, A., Ray, S., and H. Gredler, “Segment Routing Prefix SID extensions for BGP”, draft-ietf-idr-bgp-prefix-sid (work in progress), June 2016, https://datatracker.ietf.org/doc/draft-ietf-idr-bgp-prefix-sid.
- [SIGCOMM] Hartert, R., Vissicchio, S., Schaus, P., Bonaventure, O.,

Filsfils, C., Telkamp, T., Francois, P., “A Declarative and Expressive Approach to Control Forwarding Paths in Carrier-Grade Networks”, Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, DOI 10.1145/2785956.2787495, August 2015, http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p15.pdf.

---

[1] Cisco Network Services Orchestrator Enabled by Tail-f Data Sheet, http://www.cisco.com/c/en/us/products/collateral/cloud-systems-management/network-services-orchestrator/datasheet-c78-734576.html

[2] This is the Penultimate Hop Popping functionality for the MPLS data plane implementation of Segment Routing.

[3] https://www.nanog.org/meetings/nanog55/presentations/Monday/Lapukhov

3 SEGMENT ROUTING MPLS DATA PLANE

3.1 IPv4 and IPv6

Segment Routing can carry both IPv4 and IPv6 packets using the MPLS data plane. An IPv6 control plane (such as ISIS, OSPFv3, BGP) can be used to distribute Segment Routing information in the network, without dependency on IPv4. For deployments on existing IPv4-only networks, Segment Routing can also be used to carry IPv6 in MPLS with an IPv4 control plane (for 6PE or 6vPE).

3.2 Existing MPLS Data Plane

The Segment Routing architecture can be directly applied to the MPLS architecture with no change of the MPLS forwarding plane. This implies that in general only a software upgrade is required to enable basic Segment Routing functionality on a node since the existing hardware MPLS forwarding can be used for Segment Routing MPLS. In the Segment Routing MPLS implementation, a Segment Identifier is encoded as a MPLS label in the forwarding plane. A Segment List, an ordered list of segments that specifies the journey of the packet, is encoded as a stack of MPLS labels in the packet header. The segment to process, also called the Active Segment, is on top of the packet label stack. Upon completion of a segment, the related label is popped from the stack.

The MPLS architecture uses a concept of Forwarding Equivalent Class (FEC) to select the Label Switched Path (LSP) that an incoming packet must be mapped to, and which label(s), if any, must be imposed on a packet entering the MPLS network. IETF RFC 3031 defines a FEC as "a group of IP packets that are forwarded in the same manner (e.g., over the same path, with the same forwarding treatment)". When IP packets enter the MPLS network, the ingress node on the edge of the MPLS network classifies IP packets into FECs and then imposes the MPLS label(s) that correspond to that FEC.

FECs are typically created based on the destination IP prefixes in the most basic MPLS forwarding construct for IPv4/v6 traffic. This means that each prefix entry in the forwarding table is associated with a FEC. This way, a lookup in the forwarding table to find the longest prefix in the table that matches a packet's destination address, also finds the FEC of the packet, since the FEC is the longest matching prefix. The label to impose on the forwarded packet is the label associated with this longest matching prefix. But a FEC can correspond to any class of traffic that is of interest. For example source IP address, IP Precedence, etc. could be included in the classification into FECs.

If the longest prefix match of the IP destination address in the forwarding table determines the FEC, then this is equivalent to the regular destination-based IP forwarding. The label to impose is then the label associated with the longest prefix in the forwarding table that matches the IP packet's destination address. Difference with regular IP hop-by-hop forwarding and MPLS forwarding is that the classification in FECs is only done at the ingress node of the MPLS network, from then on the packet follows the path of the label.

In the Segment Routing context, the mapping of a prefix to a FEC still remains the same and the only difference is that the label corresponding to this FEC is actually derived from its Prefix-SID (signaled by IGP/BGP) instead of being signaled via traditional hop-by-hop MPLS signaling protocol like LDP.

3.3 Segment Routing Global Block

The Segment Routing Global Block (SRGB) in the Segment Routing architecture is the set of identifiers used for Global Segments. In the MPLS forwarding plane usage of Segment Routing, which is the focus of this chapter, a Segment Identifier (SID) is a label value or an index into a label block. The SID of a Local Segment is a local label value. The SID of a Global Segment is a globally unique index - a SID index. On a given node, this SID index points to a label within the SRGB of that node. The SRGB is the set of local labels that a given node reserves for Global Segments.

Each node can independently decide which range of labels it uses for Global SIDs and allocate them for its own local SRBG. The concept of global SID index and local SRGB fits the existing MPLS architecture, as the label value for a Global Segment can be managed locally, similar to other MPLS protocols.

The SID index is zero-based and points to a local label value in each node's local SRGB. For example SID index *n* points to the *n*th label in each node's local SRGB. Generally the SRGB consists of a single range of labels. In that common case, the local label value of a Global Segment be calculated by adding the SID index to the first label value in the SRGB. For example a node that allocates the label range 16000 to 23999 as SRGB finds the local label value for a Global Segment with SID index 10 by adding 10 to 16000 = 16010. 16010 is this node's local label value for SID index 10. The same Global Segment with SID index 10 has a local label value 20010 on a node that uses SRGB [20000-27999].

Since the SRGB is locally significant, each node must advertise it for the other nodes to learn about it. Other nodes need this information to calculate which label value a node expects to receive for a specific SID index. Each node must distribute its SRGB and the SID index(es) of its Global Segment(s).

Since a Global Segment's local label value is computed by adding the SID index to the SRGB base value, using the same SRGB on all nodes results in the same local label value on all nodes for a SID index. Same local label value on all nodes = global label value.

The SRGB and SID index mechanism is applicable to any type of Global Segment. The best-known Global Segments are Prefix Segments. Hence the SRGB and SID index concept applies to Prefix Segments and its variants - Node Segment and Anycast Segment.

HIGHLIGHT: SRGB and Global Prefix SID for SR MPLS

Formally, a Prefix SID is a unique index in the SRGB.

The node originating the prefix advertises the index with the prefix SID

Any node in the SR domain derives the local label associated to the Prefix SID as SRGB_base + index

The recommended deployment model calls for homogenous SRGB through the SR domain

In practice (using homogenous SRGB), each node derives the same label (same SRGB base + same index = same label) and hence the label allocated to a Prefix SID is a Global Label. For that reason, in practice, Prefix SID in SR MPLS may also refer to the Global Label allocated to the Prefix.

The SRGB does not apply to Local Segments. SIDs for Local Segments are local labels allocated from a label range outside of the SRGB.

HIGHLIGHT: Local Adjacency-SID in SR MPLS

Formally, an Adjacency-SID might be a global SID allocated from the SRGB or a local label allocated outside the SRGB.

In practice, in SR MPLS, an Adjacency-SID is a local label allocated outside the SRGB.

An Adjacency-SID is only relevant for the node that allocates it.

Other nodes of the SR domain do not install a remote Adjacency-SID in their forwarding table.

The default label range used for the SRGB on Cisco IOS XR devices is [16000-23999]. An operator can configure a non-default SRGB if desired. If not specified otherwise, this book assumes that each node uses this default SRGB. This is also the recommended deployment model.

Figure 3-1 illustrates the “same SRGB” deployment model. Node10 advertises a Prefix Segment with SID index 10 associated with its loopback prefix 1.1.1.10/32. All nodes derive their own local label value for that Prefix-SID, using their local SRGB and the received Prefix-SID index: Prefix-SID label value = SRGB Base + Prefix-SID index = 16000 + 10 = 16010. 16010 is the label value that each node expects to receive for packets on the Prefix Segment of Node10.

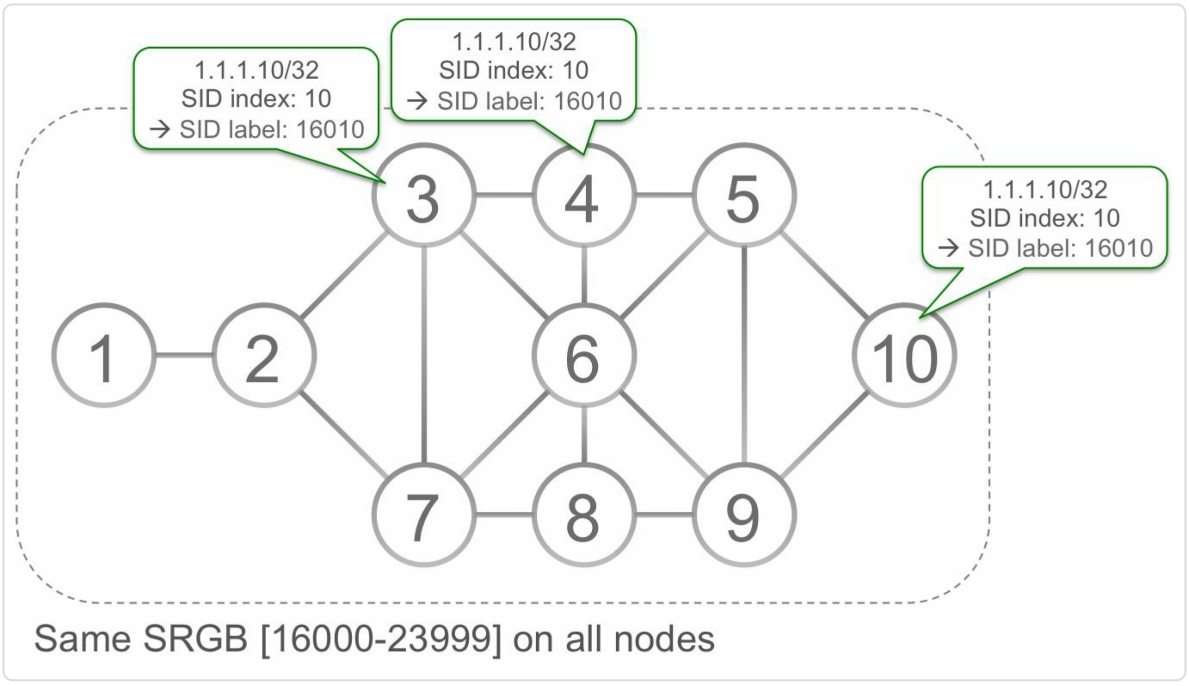


Figure 3-1: Same SRGB deployment model

In this chapter we will go on to describe the MPLS forwarding plane for Segment Routing on the basis of the same single SRGB block usage across the network. Further details on the SRGB management and other options as per the Segment Routing specifications are covered in the chapter 4.

"The operators who helped to define the SR technology insisted that the SRGB be the same across the network. All the operators that I later met confirmed that requirement.

Operating an SR domain with a consistent SRGB is not only the recommended design, it is the intended design.

The ability to use different SRGB's in the same network is like an airbag in a car: try and never use it.

To ensure that this design rule be used by the majority, we proposed a multi-vendor consensus on a default SRGB ([16000-23999])"

— Clarence Filsfils

3.4 SR MPLS Label Stack Operations

MPLS forwarding as defined in the MPLS architecture specification (RFC 3031) has three basic operations that can be performed by the MPLS network nodes, the Label Switching Routers (LSRs), on the label stack in the packet header:

PUSH

add a label on top of the label stack of the packet header

SWAP

replace the top label of the label stack of the packet header with a new label

POP

remove a label from the top of the label stack of the packet header

The Segment Routing MPLS data plane operations use these existing MPLS forwarding operations. The Segment List operations (PUSH, CONTINUE, NEXT) as described in chapter 2, "Fundamentals of Segment Routing", are mapped to the MPLS data plane operations in Table 3-1.

Table 3-1: mapping Segment List operation to MPLS label stack operation

Segment List operation	**MPLS label stack operation**
PUSH	PUSH
CONTINUE	SWAP
POP	POP

The small network topology in Figure 3-2 is used to describe the different MPLS label stack operations in the context of Segment Routing Prefix Segments. Further in this chapter we will look at the details of the MPLS forwarding entries as seen on a router running Cisco IOS XR software with Segment Routing support.

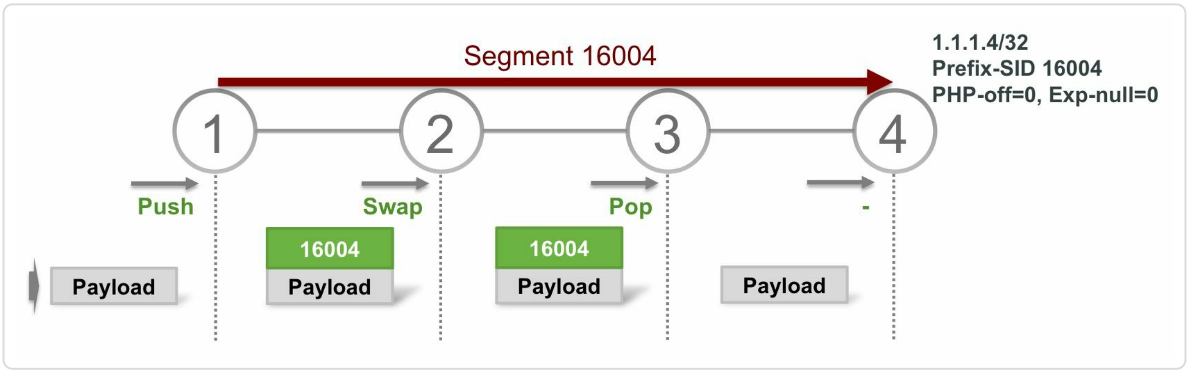


Figure 3-2: MPLS data plane operations

Segment Routing is enabled on all nodes in this network and LDP is not enabled on the nodes. All nodes in the network use the same SRGB [16000-23999]. Node4 advertises its IPv4 loopback prefix 1.1.1.4/32 along with a prefix-SID label 16004, and Node4 advertises its IPv6 loopback prefix 2001::1:1:1:4/128 along with its Prefix-SID label 17004. Since Prefix-SIDs are globally significant, all nodes in the network install the forwarding entries for these Prefix-SIDs. As the SRGB is homogenous in the SR domain, all nodes allocate the same label for the same Prefix SID. For example, all nodes allocate label 16004 as Prefix SID for 1.1.1.4/32. 16004 is a global label through the SR domain.

HIGHLIGHT

The traditional MPLS data plane is reused without any modification by using the same PUSH, POP and SWAP mechanisms for Segment List operations

3.4.1 MPLS Label Push or Imposition

A node imposes a Prefix-SID label on an incoming unlabeled IP packet if all the following conditions are satisfied:

- The IP destination address of the packet matches a prefix in the forwarding table (using longest prefix match), or the next-hop address that the IP destination resolves on (such as the BGP next-hop of a BGP destination) matches a prefix in the forwarding table. The longest matching prefix is here called “destination prefix”. The destination prefix has an associated Prefix-SID which specifies the label that must be imposed on the forwarded packet. In MPLS terms, the FEC consists of the destination prefix and all prefixes resolving on that destination prefix, and the Prefix-SID of the destination prefix is the label bound to that FEC.
- The next-hop for the destination prefix, which is the downstream neighbor towards that destination prefix in the forwarding table, is Segment Routing enabled. A non-Segment Routing next-hop node would not be able to correctly forward packets it receives with the Prefix-SID label on top.
- The node is configured to prefer Segment Routing label imposition to LDP label imposition, or the destination prefix has no associated outgoing LDP label. By default a node prefers to impose a LDP outgoing label if such label is available. If both a LDP and Segment Routing outgoing label is available for a specific prefix, and the operator wants to impose the Segment Routing label, the operator needs to explicitly configure this. Such configuration that allows for introduction of Segment Routing in existing MPLS network and interworking with LDP is described in more detail in chapter 7,

"Segment Routing and existing MPLS networks".

Figure 3-3 illustrates the MPLS label imposition operation for Segment Routing.

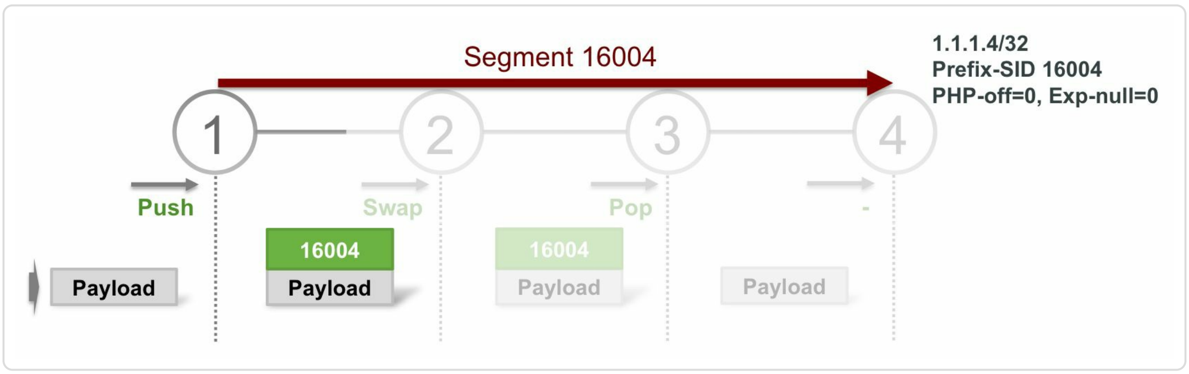


Figure 3-3: MPLS label imposition operation

Node1 has Segment Routing enabled and has no LDP enabled. Node1 has installed the forwarding entries for prefix 1.1.1.4/32, the loopback prefix of Node4, and its associated Prefix-SID 16004.

An unlabeled packet with destination IPv4 address 1.1.1.4 arrives on Node1. The forwarding lookup on Node1 for this packet finds that prefix 1.1.1.4/32 is the longest prefix matching the packet's destination address and this prefix has an associated Prefix-SID 16004 (condition 1). The next-hop towards prefix 1.1.1.4/32, Node2, is Segment Routing enabled (condition 2). None of the nodes has LDP enabled, so no outgoing LDP label is available for prefix 1.1.1.4/32 (condition 3).

All conditions for Prefix-SID label imposition are satisfied. Thus Node1 PUSHes the Prefix-SID label 16004 on the packet and steers the packet on the shortest path to Node4.

Example 3-1: MPLS label imposition IPv4 cef entry

```
RP/0/0/CPU0:xrvr-1#show cef 1.1.1.4
1.1.1.4/32, version 58, internal 0x1000001 0x81 (ptr
0xa13be574) [1], 0x0 (0xa1389878), 0xa28 (0xa1527208)
 Updated Jan 30 23:21:52.764
 local adjacency 99.1.2.2
 Prefix Len 32, traffic index 0, precedence n/a, priority 1
   via 99.1.2.2/32, GigabitEthernet0/0/0/0, 7 dependencies,
weight 0, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0xa1097250 0x0]
    next hop 99.1.2.2/32
    local adjacency
     local label 16004       labels imposed {16004}
```

The output of `show cef` in Example 3-1 shows the Cisco Express Forwarding (CEF) entry of the loopback IPv4 prefix of Node4, 1.1.1.4/32. The output shows the outgoing interface Gi0/0/0/0 and the next-hop address 99.1.2.2 for this destination prefix. This next-hop is Node2. The label that is imposed is {16004}, as highlighted in the shaded portion of the output.

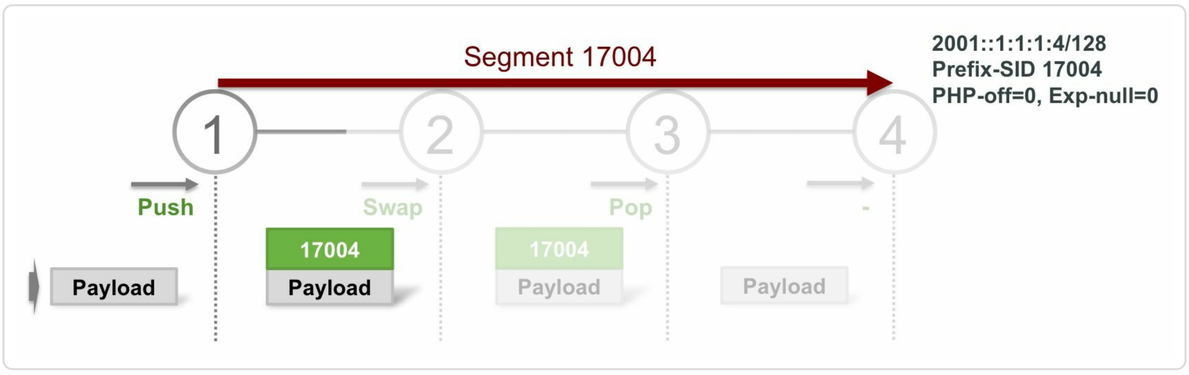


Figure 3-4: MPLS label imposition operation for IPv6

Figure 3-4 is the equivalent example for IPv6. Node1 is the ingress node and PUSHes the label 17004 on packets destined for 2001::1:1:1:4/128 and steers the traffic on the shortest path to Node4, the originator of this prefix.

Example 3-2: MPLS label imposition IPv6 cef entry

```
RP/0/0/CPU0:xrvr-1#show cef ipv6 2001::1:1:1:4
2001::1:1:1:4/128, version 1845, internal 0x1000001 0x81 (ptr
0xa12a16f4) [1], 0x0 (0xa1285b24), 0xa20 (0xa14958e8)
 Updated Jan 30 23:21:52.764
 Prefix Len 128, traffic index 0, precedence n/a, priority 1
   via fe80::f816:3eff:fe1e:55e1/128, GigabitEthernet0/0/0/0, 8
dependencies, weight 0, class 0 [flags 0x0]
     path-idx 0 NHID 0x0 [0xa1097c74 0xa1097698]
     next hop fe80::f816:3eff:fe1e:55e1/128
     local adjacency
      local label 17004      labels imposed {17004}
```

The output of `show cef ipv6` in Example 3-2 shows the CEF entry of the loopback IPv6 prefix of Node4, 2001::1:1:1:4/128. The output shows the same outgoing interface and next-hop as for the IPv4 example above. The label that is imposed for this FEC is {17004}.

3.4.2 MPLS Label Swap

Node2 in Figure 3-5 is a midpoint node on the path of the Prefix Segment from Node1 to Node4. Node2 only needs to forward the packet along the Prefix Segment path. Even if the incoming and outgoing labels are identical, Node2 uses the existing MPLS SWAP operation to forward the packet. The Prefix-SID incoming label is SWAPPED with the same Prefix-SID outgoing label due to our use of a homogenous SRGB.

In Figure 3-5, Node2 does a MPLS SWAP operation on incoming packets with top label 16004. Node2 swaps the incoming label 16004 with the same outgoing label 16004 and steers the packet on the shortest path towards 1.1.1.4/32, the loopback prefix of Node4 that is associated with Prefix-SID 16004.

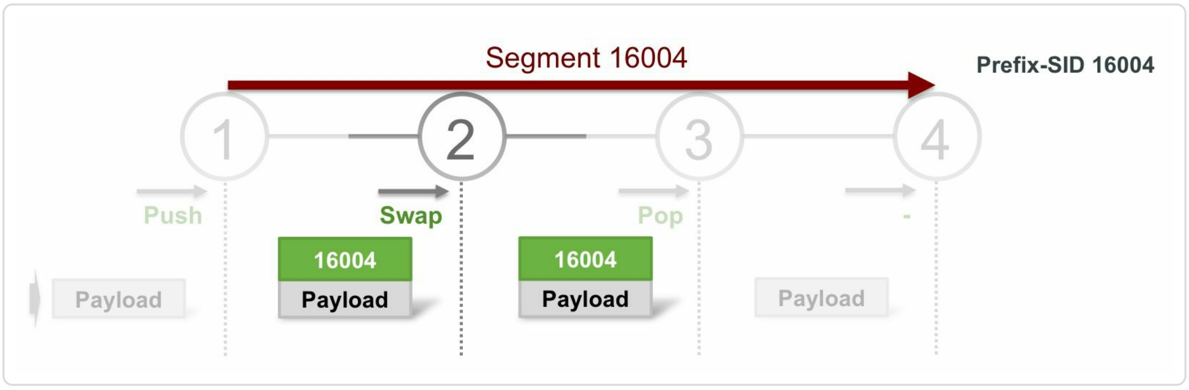


Figure 3-5: MPLS label swap operation

The output of `show mpls forwarding` on Node2 shown in Example 3-3, displays the MPLS forwarding table entry with Local Label 16004. The Outgoing Label is 16004 and the Outgoing Interface and Next Hop point to Node3, the downstream neighbor on the shortest path to 1.1.1.4/32.

Example 3-3: MPLS forwarding entry – Label swap

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 16004
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16004  16004       SR Pfx (idx 4)     Gi0/0/0/1
99.2.3.3        0
```

The equivalent case for an IPv6 Prefix Segment is nearly identical to IPv4, only the Next Hop address is the link-local IPv6 address of the next-hop in the case of IPv6 over MPLS.

The output of `show mpls forwarding` on Node2 in Example 3-4 displays the MPLS forwarding entry for local label 17004. This is the Prefix-SID label associated with IPv6 prefix 2001::1:1:1:4/128.

Example 3-4: MPLS forwarding entry for IPv6 Prefix-SID – Label swap

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 17004
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
17004  17004       SR Pfx (idx 1004)  Gi0/0/0/1
fe80::f816:3eff:fe1b:dee1    \

                                                   0
```

3.4.3 Penultimate Hop Node Behavior

Node3 is the node before the last hop node (commonly called “penultimate hop”, from Latin paenultimus, paene (“almost”) + ultimus (“last”)) on the path of the Prefix Segment from Node1 to Node4 in Figure 3-6.

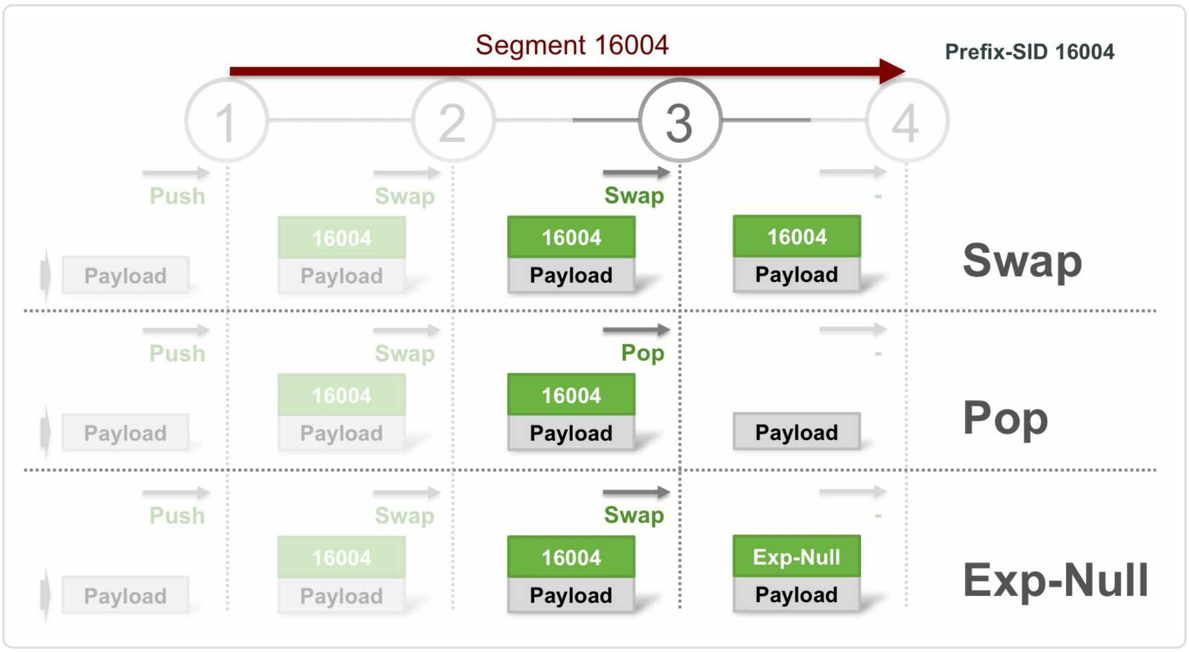


Figure 3-6: Penultimate hop behaviors

The node that originates a Prefix Segment, Node4, can specify the required behavior of the penultimate hop of that Prefix Segment, Node3, by setting or clearing flags in its Prefix-SID advertisement. The IGP control plane chapter 5 explains how this is done. The node that originates a Prefix-SID can request three possible behaviors of the penultimate hop node:

SWAP with Prefix-SID label

the penultimate hop swaps the Prefix-SID top label with the same Prefix-SID outgoing label, identical to any other midpoint node on the Prefix Segment path. In this case the ultimate node, which is the node originating the Prefix Segment, receives the packets with its own local Prefix-SID label on top.

POP

the penultimate hop removes the Prefix-SID top label before forwarding to the final node. This is the well-known "Penultimate Hop Popping" behavior.

SWAP with Explicit-null label

the penultimate hop swaps the Prefix-SID top label with the explicit-null label before forwarding to the final node.

HIGHLIGHT

The same penultimate hop handling options of traditional MPLS data plane are reused and available with Segment Routing

3.4.3.1 Swap with Prefix-SID Label

This behavior is sometimes called "Ultimate Hop Popping (UHP)" to contrast it with "Penultimate Hop Popping (PHP)". The originator of a Prefix-SID, which is the final or ultimate node of that Prefix Segment, requests to receive the packets transported on the Prefix Segment with the Prefix-SID label on top of the label stack. With this behavioral model, the final node has to do two lookups for each packet. It first has to look up the top label of the packet in its MPLS forwarding table, just to figure out that this label is its local label and must be Popped. The node then needs to do a second lookup and forward the packet to its destination based on this second lookup. If the received packet had a single label then this second lookup is an IP lookup, else if the packet had more than one label, then this second lookup would be a label lookup.

This double lookup has an impact on the forwarding performance. Similar to most other MPLS protocols, such double lookups are avoided by using the "Penultimate Hop Popping" behavior. This is described in the next section.

Cisco IOS XR nodes never request an "Ultimate Hop Popping" behavior when originating a Prefix-SID. These nodes do not install a forwarding entry for a locally originated Prefix-SID. Packets received with a local Prefix-SID label on top are dropped.

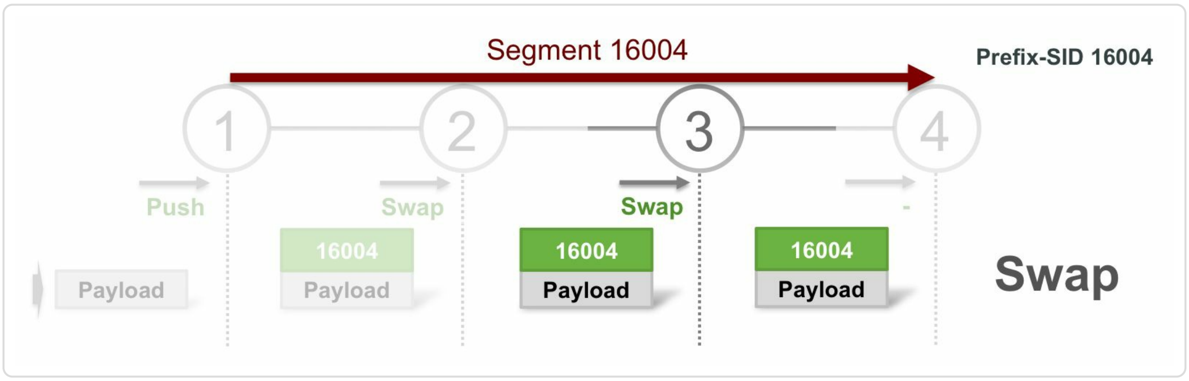


Figure 3-7: Ultimate Hop Popping

3.4.3.2 Pop Prefix-SID Label

This behavior is called "Penultimate Hop Popping" or PHP. The originator of a Prefix-SID requests its upstream neighbor nodes, which are the penultimate hop nodes, to Pop the Prefix-SID top label before forwarding the packet. Before popping the label, the penultimate hop does a (single) label lookup to find the forwarding information for the packet. With this behavior the final node only needs to do a single lookup in the forwarding table. This is the most common behavior on penultimate hop nodes and is the default behavior for Prefix Segments on Cisco IOS XR nodes.

The PHP behavior for Prefix Segments is equivalent to the behavior that is achieved by other MPLS protocols, such as LDP or BGP Labeled Unicast, where the final node advertises an implicit-Null label to its upstream neighbors. These upstream neighbors install the received implicit-Null label as a POP operation in their forwarding table. In the Segment Routing IGP control plane no implicit-null label is signaled, but the IGP control plane figures out when it is the penultimate hop and accordingly programs the implicit-null label in the forwarding entry. This is described further as part of Segment Routing IGP Control Plane operations in chapter 5.

In Figure 3-8, Node4 requests PHP functionality for the Prefix Segment it originates by clearing the PHP-off flag in the Prefix-SID advertisement. The penultimate hop Node3 will then POP the Prefix-SID label 16004 before forwarding the packet to Node4.

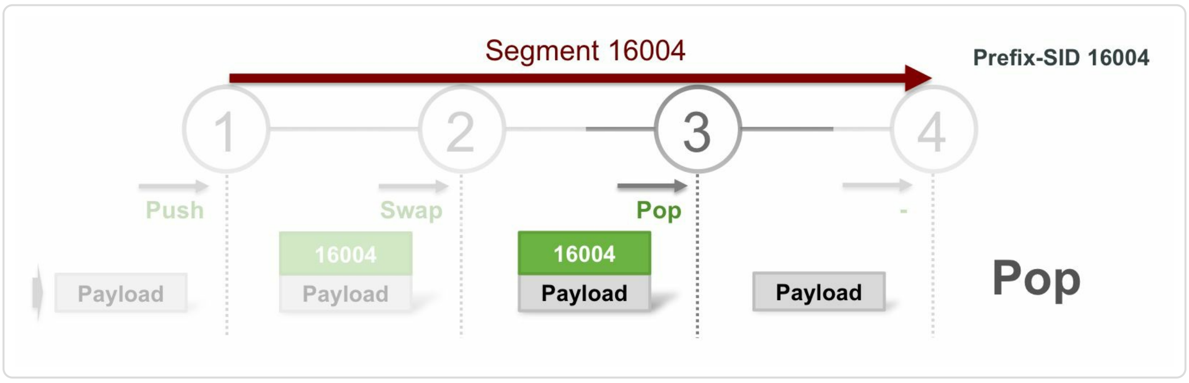


Figure 3-8: Penultimate hop popping

The output of `show mpls forwarding` on Node3 in Example 3-5 shows the MPLS forwarding table entry of Local Label 16004. The Outgoing Label is Pop and the packets are forwarded on outgoing interface Gi0/0/0/0 to next-hop 99.3.4.4, which is Node4.

Example 3-5: Penultimate hop popping MPLS forwarding entry

```
RP/0/0/CPU0:xrvr-3#show mpls forwarding labels 16004
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16004  Pop         SR Pfx (idx 4)     Gi0/0/0/0
99.3.4.4        0
```

The equivalent case for an IPv6 Prefix Segment in Example 3-6, is almost identical to the IPv4 MPLS forwarding entry, only the Next Hop address is the link-local IPv6 address of the next-hop in the case of IPv6 over MPLS.

Example 3-6: Penultimate hop popping MPLS forwarding entry for IPv6

```
RP/0/0/CPU0:xrvr-3#show mpls forwarding labels 17004
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
17004  Pop         SR Pfx (idx 1004)  Gi0/0/0/0
fe80::f816:3eff:feec:f108   \

 0
```

The use of Penultimate Hop Popping increases the forwarding efficiency. However, it has a disadvantage: the MPLS label not only contains a label value field, but also a Traffic Class (TC) field (previously known as the "experimental bits" or "EXP bits"). This TC field in the MPLS label conveys the Class of Service to be applied to the packet. If the packet arrives with one label on the penultimate hop node and that node pops the last label, then the TC field is also removed and the CoS information that was encoded in the MPLS label does not reach the ultimate node.

An operator may want to have this CoS information delivered all the way to the ultimate node for classification purposes. One possible solution is to swap the top label with an Explicit-Null label, as in the next section, instead of popping the top label.

3.4.3.3 Swap with Explicit-Null Label

If the originator of a Prefix-SID requests Explicit-null behavior, then the penultimate hop node swaps the top label with the explicit-null label before forwarding to the ultimate hop node. The ultimate hop node is the node that originated the Prefix-SID. In this case the ultimate hop node receives the packets with the Explicit-null label as top label. This way there is always a label, with the MPLS TC field, on the packet when it arrives at the final node. Consequently, the ultimate hop node can derive the CoS information of the packet from the MPLS TC field. This behavior comes at a cost since the final node has to pop the Explicit-null label and do a second lookup on the information underneath the popped label.

The Explicit-null behavior is not limited to packets with a single label. If multiple labels are on the packet's label stack, then only the top label is swapped with the explicit-null label.

The Explicit-Null label value, the actual value in the MPLS label field, is a reserved label value: the IPv4 Explicit-Null label has the value 0, the IPv6 Explicit-Null label has the value 2.

Another way to preserve the CoS information in the packet is to copy the MPLS TC field value of the top label to the IP Precedence bits of the IP header when popping the last label from the stack. Or, if multiple labels are on the stack, copy the TC field of the popped label to the TC field of the newly exposed label underneath. However, such behavior is not always desired and it is not done by default. The models used for differentiated services over MPLS are out of scope for this book, but a starting point for reading could be IETF RFC 3270.

In Figure 3-9 Node4 requests explicit-null behavior for its Prefix-SID associated with its loopback prefix 1.1.1.4/32 by setting both PHP-off and Explicit-Null flags in the Prefix-SID advertisement. The details of these flags are explained in the IGP control plane chapter 5.

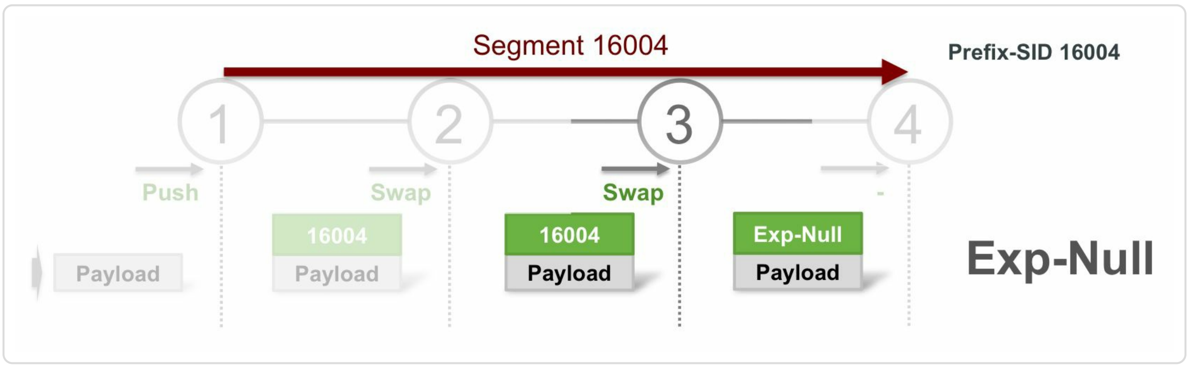


Figure 3-9: Explicit-null operation

In this case the penultimate hop, Node3, does not pop the top Prefix-SID label, but swaps the top label with the explicit-null label and forwards the packet on the shortest path towards Node4.

The packet subsequently arrives at Node4 with an explicit-null label on top of the label stack, and Node4 can first extract the MPLS “Traffic Class field” or “EXP bits” from the explicit-null label and then pop that label.

Node4 then forwards the packet based on the destination IP address or the newly exposed top label if more than one label was present.

The output of `show mpls forwarding` on Node3 in Example 3-7 displays the MPLS forwarding table entry of Local Label 16004 with Outgoing Label Exp-Null-v4 (label value 0). The Outgoing Interface and Next Hop address are on the shortest path to the destination Node4.

Example 3-7: IPv4 Explicit-Null MPLS forwarding entry

```
RP/0/0/CPU0:xrvr-3#show mpls forwarding labels 16004
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16004  Exp-Null-v4 SR Pfx (idx 4)     Gi0/0/0/0
99.3.4.4        0
```

The output of `show mpls forwarding` on Node3 in Example 3-8 displays the MPLS forwarding table entry of Local Label 17004 with Outgoing Label Exp-Null-v6. Label 17004 is the Prefix-SID label associated with an IPv6 prefix. The Outgoing Interface and Next Hop address are on the shortest path to the destination Node4.

Example 3-8: IPv6 Explicit-Null MPLS forwarding entry

```
RP/0/0/CPU0:xrvr-3#show mpls forwarding labels 17004
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
17004  Exp-Null-v6 SR Pfx (idx 1004)  Gi0/0/0/0
fe80::f816:3eff:feec:f108   \
```

0

3.4.4 Final Hop Node

Finally the destination Node4 receives the packet. Depending on the penultimate hop behavior and the number of labels on the stack, this packet is unlabeled or labeled with one or more labels on the stack. This node does a lookup on the header and forwards the packet accordingly.

3.4.5 Adjacency-SIDs and MPLS

Contrary to the different behaviors that the originating node of a Prefix Segment can request from the penultimate hop node, the MPLS label stack operation of an Adjacency Segment is always POP.

Another difference between Prefix and Adjacency Segments is that an Adjacency-SID is not associated with a prefix but with an adjacency. Therefore there is no Prefix FEC that binds to the Adjacency-SID label. No traffic will automatically get the Adjacency-SID label imposed. Adjacency Segments are used by applications that need to steer traffic over specific links, regardless of the shortest path. An example of such application is Topology Independent LFA that we will describe in chapter 9, "Topology Independent LFA (TI-LFA)". Yet another application is Segment Routing Traffic Engineering which would be covered in the next part of this book.

3.4.6 Forwarding Entry with Unlabelled

If no outgoing label is available for a prefix then an Unlabelled outgoing label is used. Unlabelled is not really an outgoing label, but is used to indicate a type of operation. The Unlabelled operation is not the same as POP and it does not mean "remove all labels". It can, for example, be observed if the downstream neighbor on the path of a Prefix Segment is not Segment Routing or LDP enabled. "Unlabelled" may indicate an error condition and a broken MPLS LSP but not always if the LSP termination was according to design.

The operation of an Unlabelled outgoing label depends on the number of paths and the label stack of the received packet. Figure 3-10 is used to illustrate the different behaviors of the Unlabelled outgoing label.

In the example network in Figure 3-10, Node11 has one path towards prefix 1.1.1.2/32: via Node1 (Gi0/0/0/0). The Prefix-SID label associated with 1.1.1.2/32 is 16002. The downstream neighbor on interface Gi0/0/0/0, Node1, is neither Segment Routing nor LDP enabled, thus the outgoing label on that path is Unlabelled.

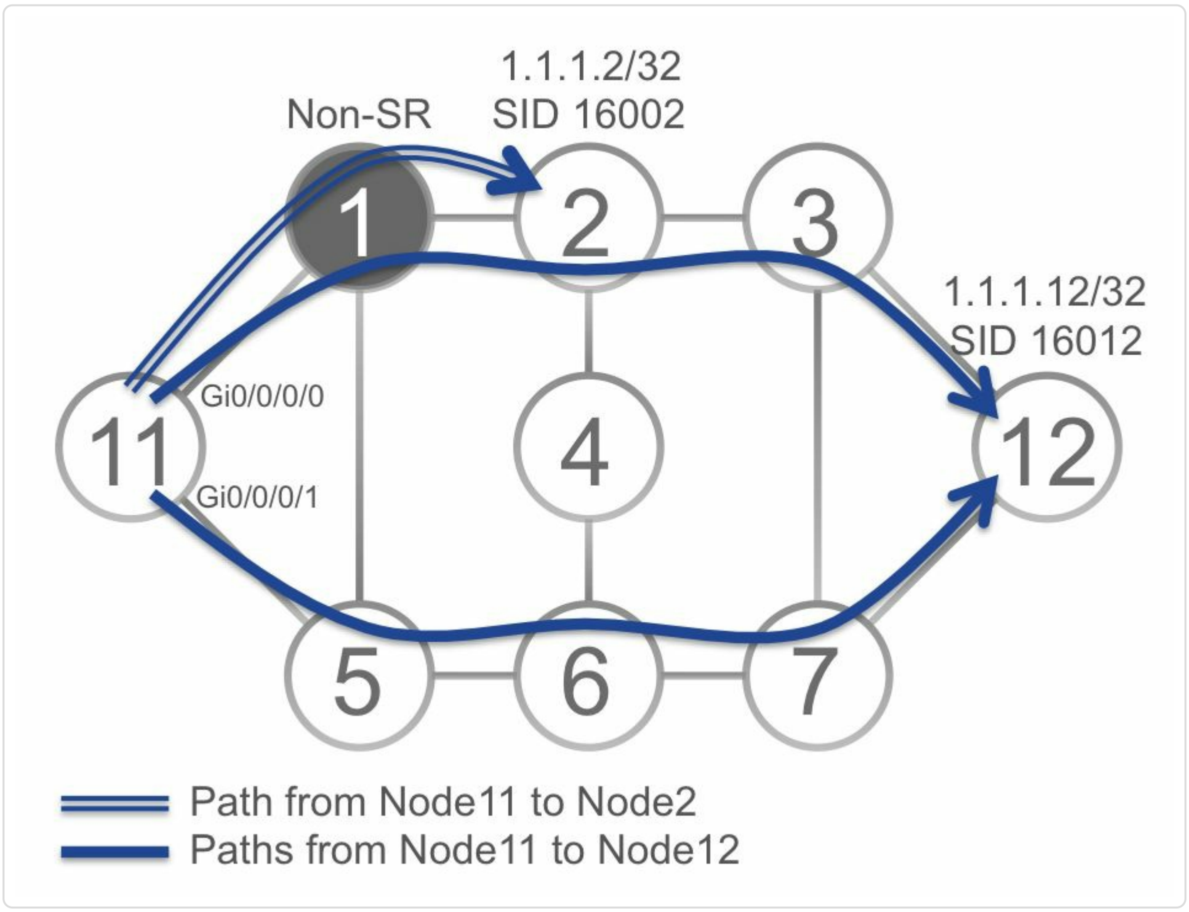


Figure 3-10: Example Unlabelled operation

The MPLS forwarding entry for the Prefix-SID label 16002 on Node11 is shown in Example 3-9.

Example 3-9: Unlabelled on single path

```
RP/0/0/CPU0:xrvr-11#show mpls forwarding prefix 1.1.1.2/32
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16002  Unlabelled  SR Pfx (idx 2)     Gi0/0/0/0
99.1.11.1       0
```

If Node11 receives an *unlabelled* packet destined for 1.1.1.2/32 then the packet is forwarded as IP packet. If the incoming packet has a *single label* with value 16002 (the label has the End of Stack (EOS) bit set to indicate it is the last label), then the label is removed and the packet is forwarded as an IP packet. If the incoming packet has more than one label and the top label is 16002 (this label has the End of Stack (EOS) bit unset to indicate there is one or more labels underneath), then the packet is dropped and this would be the erroneous termination of the LSP that we referred to previously.

In the same example network of Figure 3-10, Node11 has two equal cost paths towards prefix 1.1.1.12/32, one via Node1 (Gi0/0/0/0) and another via Node5 (Gi0/0/0/1). The Prefix-SID label associated with 1.1.1.12/32 is 16012. Since Node1 is neither Segment Routing nor LDP enabled the outgoing label on the path via Node1is Unlabelled. The downstream neighbor on interface Gi0/0/0/1, Node5, is Segment Routing enabled and the outgoing label on that path is 16012. The entry in the MPLS forwarding table for Prefix-SID label 16012 on Node11 is displayed in Example 3-10.

Example 3-10: Unlabelled in show mpls forwarding output

```
RP/0/0/CPU0:xrvr-11#show mpls forwarding prefix 1.1.1.12/32
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16012  Unlabelled  SR Pfx (idx 12)    Gi0/0/0/0
99.1.11.1       0
       16012       SR Pfx (idx 12)    Gi0/0/0/1
99.5.11.5       0
```

If Node11 receives *unlabelled* packets destined for 1.1.1.12/32 then the packets are load-balanced over the two equal cost paths. On the path via Node1 no label is imposed on the packet when forwarded (Unlabelled). On the path via Node5 a label 16012 is imposed on the packet when forwarded.

If the incoming packets have a *single label* 16012, then the node load-balances the packets over the two equal cost paths. If a packet is forwarded on the Unlabelled path, then the label is removed and the packet is forwarded as IP. If a packet is forwarded on the other path, the label is swapped with 16012 and forwarded.

If the incoming packets have *more than one label* and the top label is 16012, then the node does not load-balance the packets over the two equal cost paths but it only forwards them on the path with an outgoing label. If Node11 would load-balance over the two paths, it would mean that the traffic on the Unlabelled path would be dropped. Therefore Node11 does not use the Unlabelled path when load-balancing packets with more than one label.

The `show mpls forwarding` command has an option to display separate entries for EOS0 packets (packets with >1 labels) and EOS1 packet (packets with 1 label). The EOS 0 or 1 is displayed in the last column. Looking at the last column of the output in Example 3-11, there is a single path for EOS0 packets, the path via Gi0/0/0/1. For EOS1 packets the two paths are available.

Example 3-11: EOS0 and EOS1 mpls forwarding entries

```
RP/0/0/CPU0:xrvr-11#show mpls forwarding labels 16012 both-eos
Local  Outgoing    Outgoing     Next Hop        Bytes
EOS
Label  Label       Interface                    Switched
------ ----------- ------------ --------------- ------------ --
-
16012  16012       Gi0/0/0/1    99.5.11.5       0            0
       Unlabelled  Gi0/0/0/0    99.1.11.1       0            1
       16012       Gi0/0/0/1    99.5.11.5       0            1
```

Example 3-12 and Example 3-13 illustrate the unlabeled prefix path in the routing table and CEF table.

Example 3-12: Example of an unlabelled path in RIB

```
RP/0/0/CPU0:xrvr-11#show route 1.1.1.12/32 detail

Routing entry for 1.1.1.12/32
  Known via "ospf 1", distance 110, metric 41, labeled SR, type
intra area
  Installed Feb 15 19:39:35.286 for 00:00:48
  Routing Descriptor Blocks
    99.1.11.1, from 1.1.1.12, via GigabitEthernet0/0/0/0
      Route metric is 41
      Label: None
```

```
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        Path id:1       Path ref count:0
        NHID:0x1(Ref:20)
        OSPF area: 0
      99.5.11.5, from 1.1.1.12, via GigabitEthernet0/0/0/1
        Route metric is 41
        Label: 0x3e8c (16012)
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        Path id:2       Path ref count:0
        NHID:0x2(Ref:50)
        OSPF area: 0
    Route version is 0x18f (399)
    Local Label: 0x3e8c (16012)
    IP Precedence: Not Set
    QoS Group ID: Not Set
    Flow-tag: Not Set
    Fwd-class: Not Set
    Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD
  Type RIB_SVD_TYPE_LOCAL
    Download Priority 1, Download Version 3820
    No advertising protos.
```

Example 3-13: Example of an unlabeled path in CEF

```
  RP/0/0/CPU0:xrvr-11#show cef 1.1.1.12/32
  1.1.1.12/32, version 3820, internal 0x1000001 0x81 (ptr
  0xa13f2df4) [1], 0x0 (0xa13d7ea8), 0xa28 (0xa174212c)
   Updated Feb 15 19:39:57.504
   local adjacency 99.1.11.1
   Prefix Len 32, traffic index 0, precedence n/a, priority 1
     via 99.1.11.1/32, GigabitEthernet0/0/0/0, 15 dependencies,
  weight 0, class 0 [flags 0x0]
      path-idx 0 NHID 0x0 [0xa10b12a4 0x0]
      next hop 99.1.11.1/32
      local adjacency
       local label 16012      labels imposed {None}
     via 99.5.11.5/32, GigabitEthernet0/0/0/1, 10 dependencies,
  weight 0, class 0 [flags 0x0]
      path-idx 1 NHID 0x0 [0xa10b13f4 0x0]
      next hop 99.5.11.5/32
      local adjacency
       local label 16012       labels imposed {16012}
```

3.4.7 IGP SR MPLS Forwarding Entries

The network topology in Figure 3-11 is used to illustrate the various possible Segment Routing MPLS forwarding entries. All nodes in this network have Segment Routing enabled. No LDP is enabled on the nodes. All IGP link metrics are equal and all nodes use the same SRGB [16000-23999]. The network is dual stack: IPv4 and IPv6, using ISIS as IGP. All nodes are Level-2 nodes. The choice of the IGP does not play a role in the MPLS forwarding entries; they are the same regardless of the IGP programming them. Each node advertises its loopback prefix with an associated Prefix-SID. In general each node requests the default PHP behavior for its Prefix-SIDs, except Node4. Node4 requests to receive packets on its Prefix Segment with Explicit-null as top label.

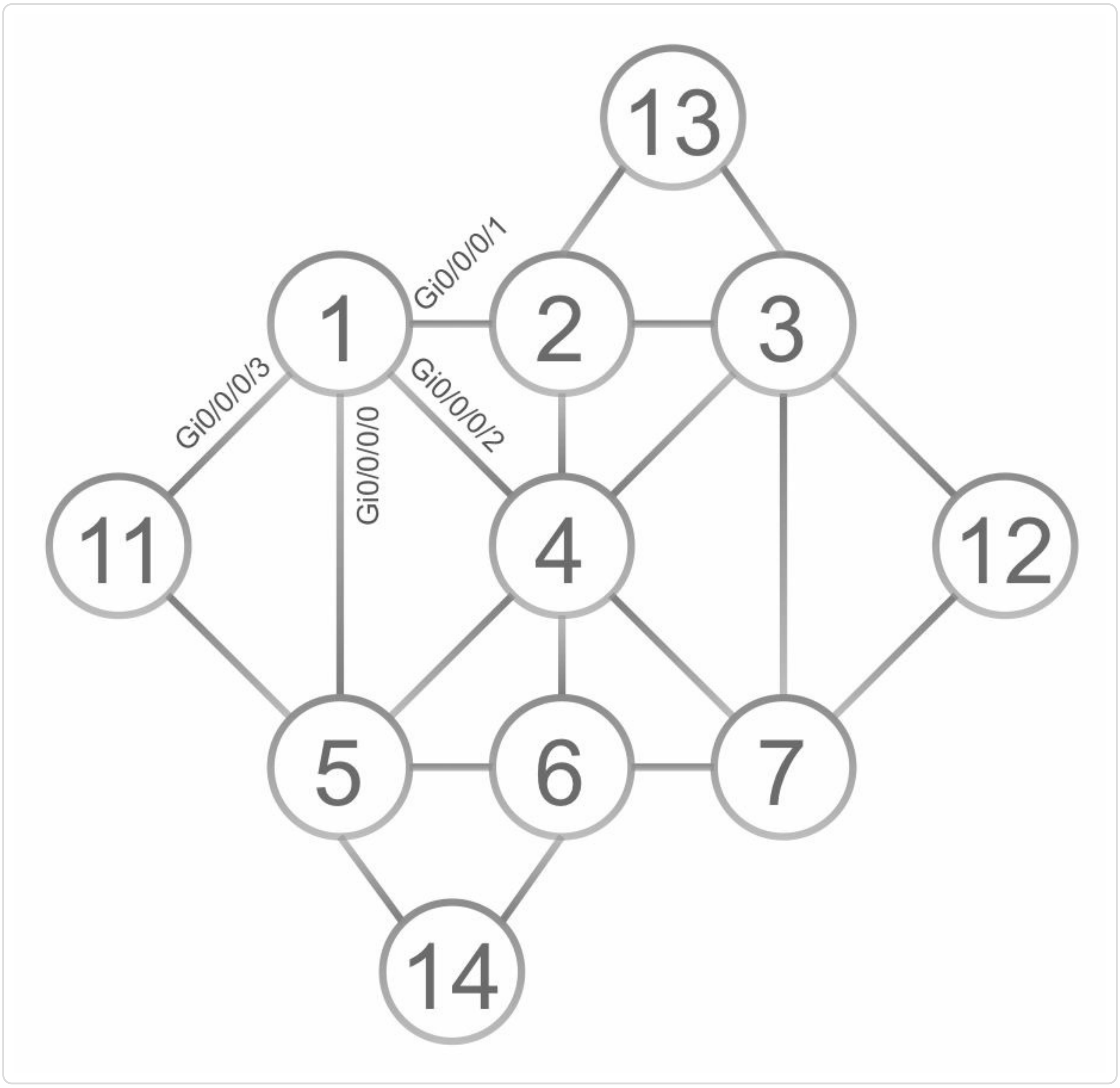


Figure 3-11: Network topology to illustrate the SR MPLS forwarding entries

Example 3-14 shows the output of `show mpls forwarding` as collected on Node1 in the topology of Figure 3-11. This command shows the MPLS forwarding table, or LFIB. To simplify the output, no fast-reroute local protection is enabled on any of Node1's interfaces. A display output filter (`| excl "fe80" | incl "[-A-Z]"`) is used to only display the IPv4 MPLS forwarding entries. It excludes lines with "fe80" which are the lines with IPv6 link local address and then includes "[-A-Z]" to drop the lines that only contain “0” (see Example 3-15 for an example of such lines).

Example 3-14: IPv4 SR MPLS forwarding entries on Node1

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding | excl "fe80" | incl
"[-A-Z]"
Local  Outgoing    Prefix             Outgoing     Next
Hop          Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ---------
------ ------------
16002  Pop         SR Pfx (idx 2)     Gi0/0/0/1
99.1.2.2        0
16003  16003       SR Pfx (idx 3)     Gi0/0/0/1
99.1.2.2        0
16003              SR Pfx (idx 3)     Gi0/0/0/2
99.1.4.4        0
16004  Exp-Null-v4 SR Pfx (idx 4)     Gi0/0/0/2
99.1.4.4        0
16005  Pop         SR Pfx (idx 5)     Gi0/0/0/0
99.1.5.5        0
16006  16006       SR Pfx (idx 6)     Gi0/0/0/0
99.1.5.5        0
16006              SR Pfx (idx 6)     Gi0/0/0/2
99.1.4.4        0
16007  16007       SR Pfx (idx 7)     Gi0/0/0/2
99.1.4.4        0
16011  Pop         SR Pfx (idx 11)    Gi0/0/0/3
99.1.11.11      0
16012  16012       SR Pfx (idx 12)    Gi0/0/0/1
99.1.2.2        0
16012              SR Pfx (idx 12)    Gi0/0/0/2
99.1.4.4        0
16013  16013       SR Pfx (idx 13)    Gi0/0/0/1
99.1.2.2        0
16014  16014       SR Pfx (idx 14)    Gi0/0/0/0
99.1.5.5        0
24000  Pop         SR Adj (idx 1)     Gi0/0/0/3
99.1.11.11      0
24001  Pop         SR Adj (idx 3)     Gi0/0/0/3
99.1.11.11      0
```

```
24002   Pop         SR Adj (idx 1)      Gi0/0/0/1
99.1.2.2        0
24003   Pop         SR Adj (idx 3)      Gi0/0/0/1
99.1.2.2        0
24004   Pop         SR Adj (idx 1)      Gi0/0/0/0
99.1.5.5        0
24005   Pop         SR Adj (idx 3)      Gi0/0/0/0
99.1.5.5        0
24006   Pop         SR Adj (idx 1)      Gi0/0/0/2
99.1.4.4        0
24007   Pop         SR Adj (idx 3)      Gi0/0/0/2
99.1.4.4        0
```

The output of the command is numerically sorted on the first column, the Local Label value. The forwarding entries in the beginning of the output – with local labels in the 16000 range – are the Prefix-SID labels of the remote nodes in the topology. Remember these Prefix-SID labels are allocated from the SRGB, label range [16000-23999] in this case. The Prefix-SID label of each node is 16000 plus the Prefix-SID index, with the index equal to the number of the node in this example (e.g. Node2 uses Prefix-SID index 2).. The Prefix-SID index is also displayed in the third column of the `show mpls forwarding` output as `SR Pfx (idx` *`n`*`)` where *n* is the Prefix-SID index.

The forwarding entries at the bottom of the output – with local labels in the 24000 range and with the third column containing `SR Adj (idx n)` – are the adjacency-SID labels, locally allocated from the dynamic label range for each of Node1's four adjacencies. Two Adjacency-SIDs are allocated for each adjacency, one protectable and one unprotected. The details of these two types of Adjacency-SIDs are discussed as part of the IGP Control Plane in chapter 5. For now it is enough to know that each adjacency counts two Adjacency-SIDs. The forwarding instruction for Adjacency-SIDs is to *pop the top label and send the packet out of the* *adjacency link*. The index "idx *n*" in the third column should not be confused with the Prefix-SID index and are internal IGP index representations which can be ignored.

Refer to the network topology in Figure 3-11 the output collected on Node1 in Example 3-14 for the following discussions.

Node2 is adjacent to Node1, the node where the output is collected. The shortest path from Node1 to Node2 is via the link connecting the two nodes. Therefore is Node1 a penultimate hop for packets destined for Node2. Node2 advertises its prefix-SID 16002 with the flags in the Prefix-SID advertisement set to request the default Penultimate Hop Popping functionality. Packets arriving at Node1 with top label 16002 are forwarded to Node2 with the top label popped. This is shown in line 5 of Example 3-14.

From Node1 to Node3 there are two equal cost paths: one via Node2 and another via Node4. Packets arriving on Node1 with top label 16003, which is the label of the Prefix-SID associated with the loopback prefix of Node3, are load-balanced over the two equal cost paths, via Node2 and via Node4. Because all nodes in the network use the same SRGB, the local label and the outgoing label for both paths is the same: Prefix-SID label 16003. This is shown in lines 6 and 7 of Example 3-14.

Node4 is an adjacent node of Node1, and the shortest path from Node1 to Node4 is via the link between the nodes. Therefore is Node1 a penultimate hop for packets destined for Node4. Node4 advertises its prefix-SID 16004 with the flags in the Prefix-SID advertisement set to request the Explicit-null behavior for this Prefix-SID. Packets arriving at Node1 with top label 16004 are forwarded to Node4 with the top label swapped with the

Explicit-null label. This is shown in line 8 of Example 3-14. Note that the label value of the Explicit-null label depends on the address-family. For IPv4 the Explicit-null label has value 0 and for IPv6 the label value is 2.

The other entries in the MPLS forwarding table are equivalent to the ones just described.

Labels 16005 and 16011 are the Prefix-SID labels for Node5 and Node11. These nodes are adjacent to Node1 and they requested the default PHP behavior, thus the outgoing label is Pop. See lines 9 and 13 of Example 3-14.

Labels 16006 and 16012 are the Prefix-SIDs of Node6 and Node12. These nodes are not adjacent to Node1 and there are two equal cost paths towards each of these nodes, so there are two outgoing label entries, one for each path. The incoming Prefix-SID label is swapped with the same outgoing label for both paths. See lines 10-11 and 14-15 of Example 3-14.

Labels 16007, 16013, and 16014 are the Prefix-SID labels for respectively Node7, Node13, and Node14. These nodes are not adjacent to Node1 and there is a single path towards each of these nodes. The incoming Prefix-SID label is swapped with the same outgoing label.

All Adjacency-SID entries use the MPLS Pop operation.

If for example a packet arrives at Node1 with top label 24000, which is an Adjacency-SID associated with the adjacency to Node11, the label is popped and the packet is forwarded over the link of the adjacency, interface Gi0/0/0/3 in this example. See line 18 of Example 3-14.

HIGHLIGHT

With Segment Routing and use of homogenous SRGB, the MPLS forwarding entries on any router in the network are greatly simplified and it is much easier to correlate them to their IPv4/IPv6 prefix destinations.

Global Labels for Prefix Segments also make troubleshooting easier across the network.

Example 3-15 shows the output of `show mpls forwarding` for the IPv6 MPLS forwarding entries. The IPv6 topology of the example is congruent with the IPv4 topology, so the discussion of the IPv4 Prefix-SIDs can easily be translated to the IPv6 Prefix-SIDs.

Example 3-15: IPv6 SR MPLS forwarding entries on Node1

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
17002  Pop         SR Pfx (idx 1002)  Gi0/0/0/1
fe80::f816:3eff:fe1e:55e1    \

 0
17003  17003       SR Pfx (idx 1003)  Gi0/0/0/1
 fe80::f816:3eff:fe1e:55e1    \

 0
       17003       SR Pfx (idx 1003)  Gi0/0/0/2
fe80::f816:3eff:feae:5125    \

   0
17004  Exp-Null-v6 SR Pfx (idx 1004)  Gi0/0/0/2
fe80::f816:3eff:feae:5125    \

 0
17005  Pop         SR Pfx (idx 1005)  Gi0/0/0/0
fe80::f816:3eff:fe1d:edf4    \
 0
17006  17006       SR Pfx (idx 1006)  Gi0/0/0/0
fe80::f816:3eff:fe1d:edf4   \

 0
       17006       SR Pfx (idx 1006)  Gi0/0/0/2
fe80::f816:3eff:feae:5125   \

 0
17007  17007       SR Pfx (idx 1007)  Gi0/0/0/2
fe80::f816:3eff:feae:5125   \

                                         0
17011  Pop         SR Pfx (idx 1011)  Gi0/0/0/3
fe80::f816:3eff:fe29:1f52   \

 0
17012  17012       SR Pfx (idx 1012)  Gi0/0/0/1
fe80::f816:3eff:fe1e:55e1   \

 0
       17012       SR Pfx (idx 1012)  Gi0/0/0/2
fe80::f816:3eff:feae:5125   \

          0
17013  17013       SR Pfx (idx 1013)  Gi0/0/0/1
fe80::f816:3eff:fe1e:55e1   \

 0
17014  17014       SR Pfx (idx 1014)  Gi0/0/0/0
fe80::f816:3eff:fe1d:edf4   \

 0
24008  Pop         SR Adj (idx 1)     Gi0/0/0/1
fe80::f816:3eff:fe1e:55e1   \

 0
24009  Pop         SR Adj (idx 3)     Gi0/0/0/1
fe80::f816:3eff:fe1e:55e1   \

 0
24010  Pop         SR Adj (idx 1)     Gi0/0/0/2
fe80::f816:3eff:feae:5125   \
                                                        0
24011  Pop          SR Adj (idx 3)     Gi0/0/0/2
fe80::f816:3eff:feae:5125   \

 0
24012  Pop          SR Adj (idx 1)     Gi0/0/0/0
fe80::f816:3eff:fe1d:edf4   \

 0
24013  Pop          SR Adj (idx 3)     Gi0/0/0/0
fe80::f816:3eff:fe1d:edf4   \

                  0
24014  Pop          SR Adj (idx 1)     Gi0/0/0/3
fe80::f816:3eff:fe29:1f52   \

 0
24015  Pop          SR Adj (idx 3)     Gi0/0/0/3
fe80::f816:3eff:fe29:1f52   \

 0
```

3.5 MPLS TTL and TC (or EXP) Treatment

An MPLS label contains besides a 20 bits Label Value also a 3 bits Traffic Class (TC) field and a 8 bits Time To Live (TTL) field. See Figure 3-12. The Bottom of Stack flag (S) is set if the label is the last entry in the label stack. The TC field was formerly known as “MPLS Experimental bits”, or simply “EXP bits”[1], and is used to specify the Class of Service the packet requires. The use of the MPLS TTL field in the label is equivalent with the use of the TTL field in the IPv4 header or the Hop Limit field in the IPv6 header. To simplify this book will refer to both the IPv4 Time To Live field and the IPv6 Hop Limit field as “IP TTL”, unless explicitly specified differently.

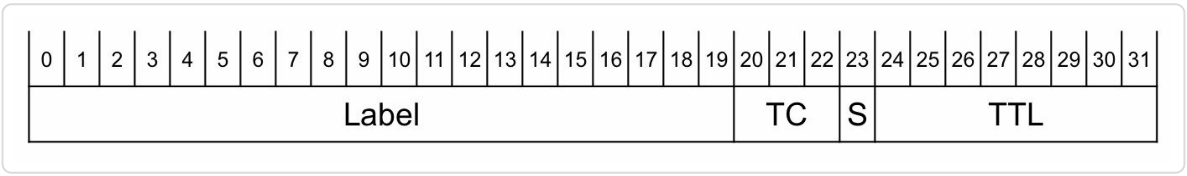


Figure 3-12: MPLS label format

With:

- Label: Label Value, 20 bits
- TC: Traffic Class, 3 bits
- S: Bottom of Stack, 1 bit
- TTL: Time to Live, 8 bits

This section explains behaviors of the existing traditional MPLS data plane which is similarly applicable in the context of the Segment Routing MPLS data plane.

3.5.1 TTL Processing IP-to-MPLS and MPLS-to-IP

When an ingress node of the Segment Routing MPLS Domain receives an IP packet that matches a prefix with an associated Prefix-SID, it imposes the Prefix-SID MPLS label and forwards the labeled packet towards its destination. Depending on its TTL propagation configuration, the ingress node either copies the IP TTL value of the incoming IP packet to the MPLS TTL field of the MPLS label after decrementing it by one, or the node sets the MPLS TTL field to 255. The default behavior is to copy the IP TTL value to the MPLS TTL field. In the case multiple labels are imposed (e.g. for Traffic Engineering, TI-LFA protection, or L3VPN), all imposed labels get the same MPLS TTL value, copied from the IP TTL field. See Figure 3-13.

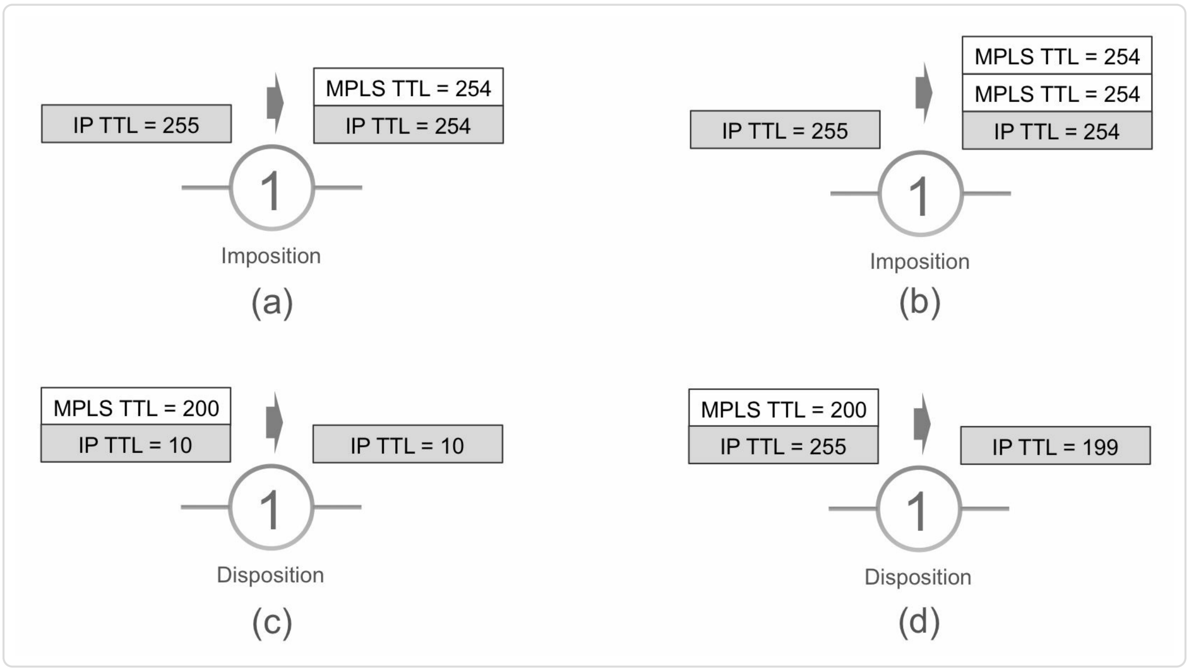


Figure 3-13: IP TTL propagation

Configuring `mpls ip-ttl-propagation disable` changes the default behavior. This command stops the copying of the IP TTL to the

MPLS TTL fields of the imposed labels and the MPLS TTL fields of all imposed labels are set to 255 regardless of the IP TTL value in the received IP packet.

This configuration command has two additional options to disable the copying of the IP TTL field only for locally generated packets (`local`) or only for forwarded packets (`forwarded`). If no option is specified then the command applies to both types of traffic.

When the last label of the packet's label stack is popped, and the IP header is exposed again, the default behavior is to copy the MPLS TTL value of the label of the received packet to the IP TTL field after decrementing it by one. However, this only happens if the MPLS TTL value of the received packet is smaller than the IP TTL value of the IP packet underneath. This condition is to avoid eternally looping packets. If a larger MPLS TTL value would be copied to the IP TTL field and the IP packet would be re-injected in the MPLS network – possibly due to a routing loop – the IP TTL would never decrement to zero and the packet would loop "forever".

If `mpls ip-ttl-propagation disable` is configured, then the MPLS TTL value is not copied to the IP TTL field. As a result, all the hops traversed by the packet in the MPLS network are unaccounted for.

Service Providers often use `mpls ip-ttl-propagation disable` to hide their internal MPLS network from traceroute probes. With `mpls ip-ttl-propagation disable` configured, the MPLS network appears as a single hop in a traceroute.

3.5.2 TTL Processing MPLS-to-MPLS

The operations on the MPLS TTL fields that are used when incoming and outgoing packets are both labeled are equivalent to the operations described in the previous section. See Figure 3-14.

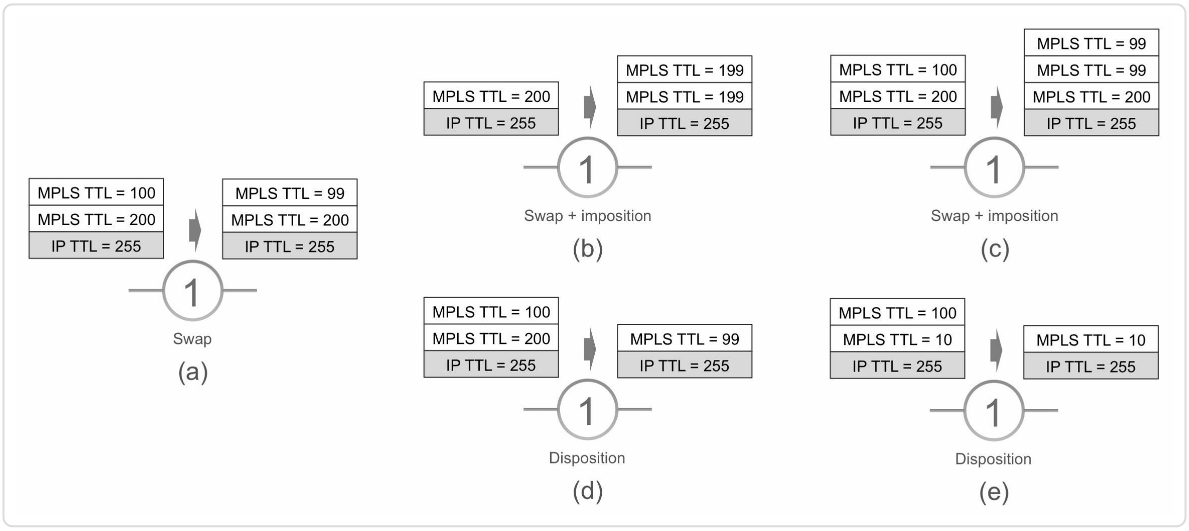


Figure 3-14: MPLS TTL propagation

If the top label of an incoming packet is swapped, then the MPLS TTL value of the top label of the incoming packet is decremented by one and copied to the MPLS TTL field of the swapped label. See Figure 3-14 (a).

If the top label of an incoming packet is swapped and one or more additional labels are pushed on the packet, then the MPLS TTL value of the top label of the incoming packet is decremented by one and copied to the MPLS TTL field of the swapped label and all newly imposed labels. See Figure 3-14 (b) and (c).

If the top label of an incoming packet is popped, then the MPLS TTL value of the received packet is decremented by one and copied to the MPLS TTL field of the newly exposed label. See Figure 3-14 (d). However, this only happens if the MPLS TTL value is smaller than the MPLS TTL value of the newly exposed label. Otherwise, the (larger)

MPLS TTL value is not copied to the newly exposed label, as illustrated in Figure 3-14 (e). This behavior prevents ever-lasting packet loops as described in the previous section.

3.5.3 Handling MPLS TTL Expiry

When the IP TTL field value of an IP packet reaches 0 (TTL "expires"), then the node where the TTL expired sends a "Time exceeded" ICMP message (ICMP type 11, code 1) to the source address of the packet, including a part of the original packet. This is well known behavior.

Using ICMP in an MPLS network may require some changes to its behavior. ICMP has been extended by IETF RFC 4950 to provide more information when used in an MPLS network.

When a node receives an MPLS packet that is not deliverable to its destination for some reason, the node strips off the MPLS header. If the underlying packet is an IP packet, then the node generates an ICMP message indicating the reason the packet could not be delivered. This ICMP message is sent to the source IP address of the undeliverable packet. If the MPLS encapsulated packet is not an IP packet, then no ICMP message is sent.

The ICMP message contains the IP header and part of the payload of the undeliverable packet. If the node uses the ICMP extension specified in IETF RFC 4950, then the ICMP message also includes the MPLS label stack of the undeliverable packet, exactly as it was when the packet arrived on the node. The MPLS label stack of the packet is important as it may point to the reason why the packet was not deliverable.

The ICMP message is sent to the source IP address of the IP packet encapsulated in the MPLS header. This source IP address may have no meaning to the node generating the ICMP message, and the node may find itself in a situation where it is unable to route the ICMP message back to the source. This problem will often occur, for example if MPLS is used to provide VPN services where the VPN traffic is tunneled over the network using MPLS. Therefore instead of directly sending the ICMP message to the source, the node uses the label stack of the original undeliverable packet, imposes that label stack on the ICMP packet, and forwards the packet according to the top label of this newly imposed label stack. This way the ICMP packet first travels downstream to the end of the Label Switched Path using its label stack, and then from there travels to the IP destination of the ICMP message, which is the source IP address of the undeliverable packet. IETF RFC 3032 describes this mechanism. See illustration in Figure 3-15: send ICMP message for labeled packet. The labeled packet travelling from Node1 to Node4 has a TTL=1 when it arrives on Node2. Since the TTL expires on Node2, Node2 drops the packet and generates an ICMP time exceeded message to report the error to the source node of the dropped packet. Node2 uses the ICMP extension of RFC 4950 to include the label stack of the dropped packet in the ICMP message. According to the procedure in RFC 3032, Node2 can now add the label stack of the incoming packet (but with an increased TTL) on the generated ICMP message and forward the packet according to this label stack. This causes the ICMP message to first travel all the way to Node4, where it is forwarded back to Node1 based on the destination address of the ICMP message (99.1.2.1).

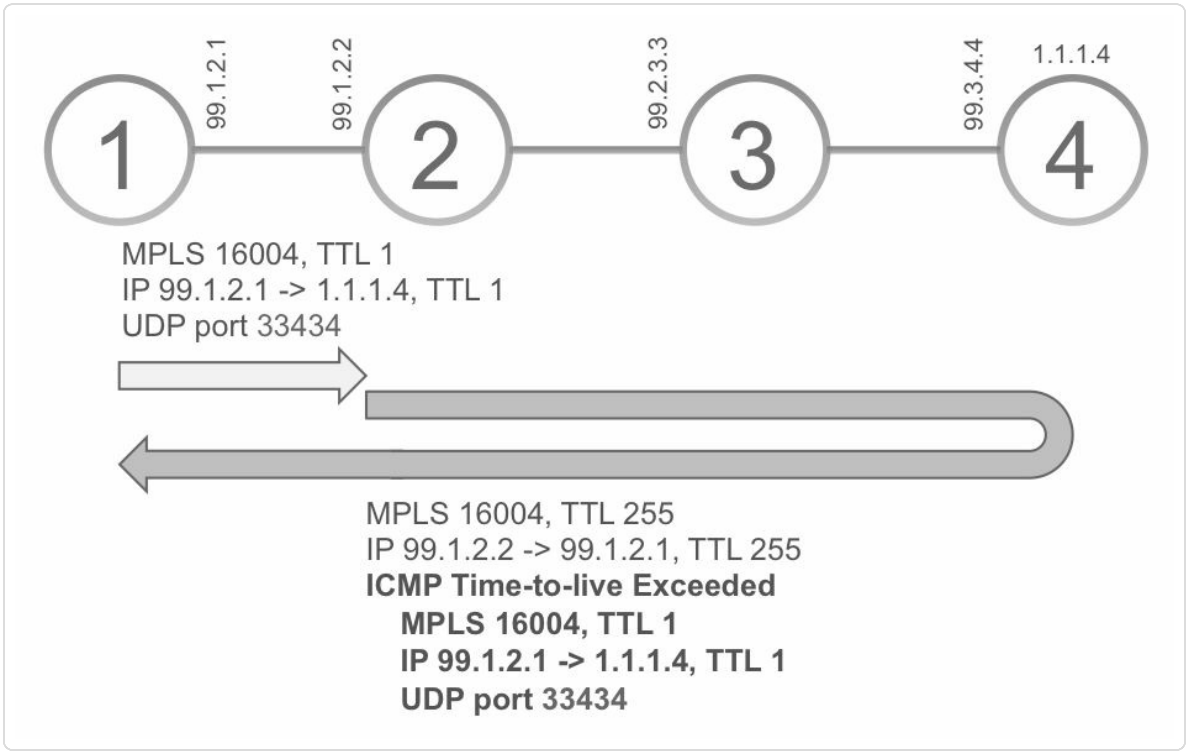


Figure 3-15: send ICMP message for labeled packet

By default, Cisco IOS XR uses the above procedure when generating and sending an ICMP message along the original LSP for an MPLS encapsulated packet. The default behavior can be overridden on an individual node by configuring `mpls ip ttl-expiration pop <n>`, with n the number of labels on the stack. When a packet is undeliverable and the number of labels in the label stack of the packet does not exceed the configured number of labels "n", then the generated ICMP message is forwarded by regular IP forwarding. If the number of labels exceeds the configured number "n", then the default behavior to forward along the original LSP is applied. If using the round trip delay information as provided by the traceroute command is of importance, then this functionality could be used. With the default behavior the round trip delay includes the time to go all the way to the end of the LSP and back, which may not be desirable. The above configuration must then be applied to all nodes in the network.

The use of these ICMP mechanisms is used for troubleshooting MPLS forwarding plane using traceroute operation and the same is also applicable to Segment Routing. These procedures are described in more detail in the OAM chapter 11, “Verifying connectivity in SR MPLS network”.

HIGHLIGHT

The TTL processing and propagation mechanisms of traditional MPLS forwarding plane also apply similarly to Segment Routing

3.5.4 TC or EXP Field Processing

This book does not cover details of QoS or DiffServ, but only touches on the very basic QoS header fields and how they are handled in the MPLS data plane. The Class of Service of an IP packet can either be set in the three bits IP Precedence field or in the six bits DiffServ Codepoint field (DSCP). Initially, only the Precedence part (3 bits) of the Type of Service field (TOS) in the IP header was used for QoS. Later DiffServ QoS was introduced (IETF RFC 2474 and IETF RFC 2475), increasing the number of bits in the IP header that could be used for QoS to six. Figure 3-16 uses the Class Selector (CS) DSCPs, which map directly to the Precedence bits. Class Selector DSCPs are values that are backward compatible with IP precedence. For example CS1 has the same bits set in the IP header as Precedence 1.

An MPLS label also has a field that is used for QoS: the Traffic Class (TC) field, formerly known as the “EXP bits”. Figure 3-16 illustrates the procedures used for the QoS fields for the different MPLS label stack operations.

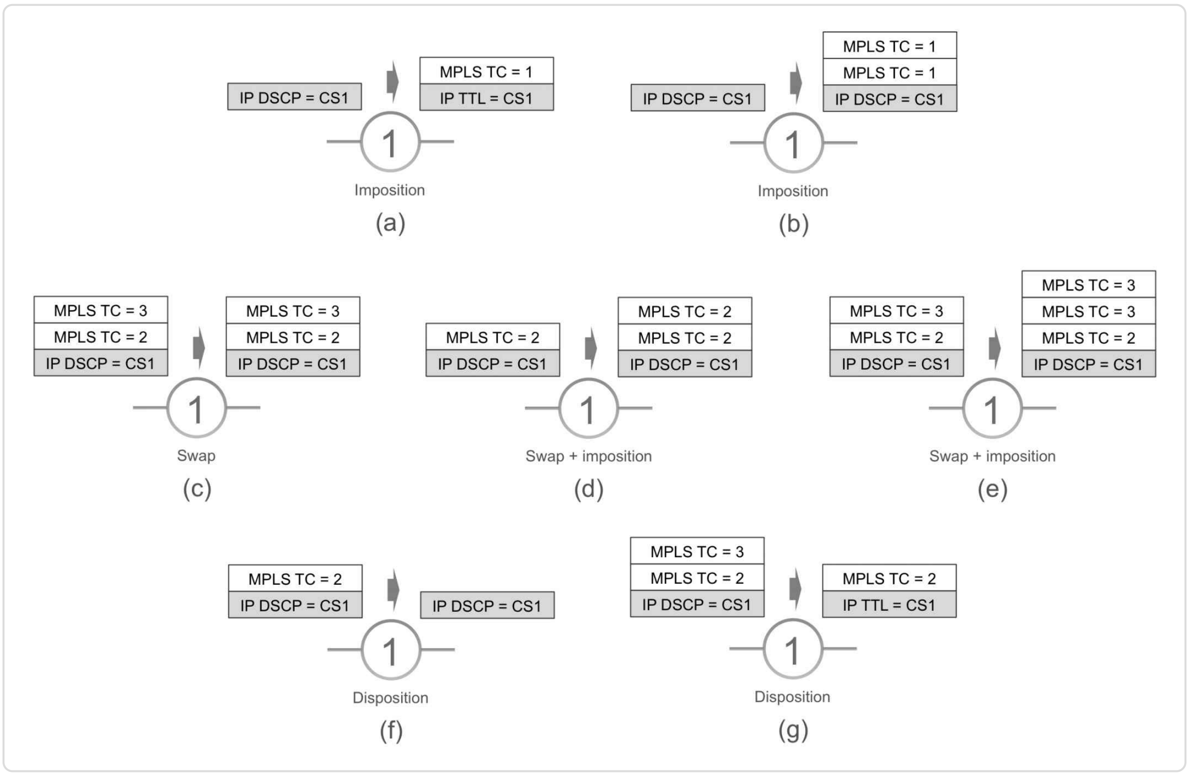


Figure 3-16: IP TOS propagation

When *imposing* one or more labels on an incoming IP packet, the default behavior in Cisco IOS XR is to copy the Precedence bits in the IP header to the MPLS TC field of all imposed labels. If the six bits of the DSCP field in the IP header are used, only the first three bits of DSCP are copied to the MPLS TC field of the imposed labels. See Figure 3-16 (a) and (b).

When *popping* the only label of an incoming labeled packet, the TC field of the received label is by default *not* copied to the newly exposed IP header’s Precedence bits or DSCP field. See Figure 3-16 (f).

If the top label of an incoming packet is *swapped*, then the received MPLS TC field is copied to the MPLS TC field of the swapped label. See Figure 3-16 (c).

If the top label of an incoming packet is swapped and one or more additional labels are *imposed* on the packet, then the received MPLS TC field of the top label is copied to the MPLS TC field of the swapped label and all newly imposed labels. See Figure 3-16 (d) and (e).

If the top label of an incoming packet is *popped*, then the received MPLS TC field is by default *not* copied to the MPLS TC field of the newly exposed label. See Figure 3-16 (g).

This is the default behavior. Different behaviors are possible by using MQC configurations. The details of such configuration are out of the scope for this book.

HIGHLIGHT

Segment Routing with MPLS forwarding plane uses the same mechanism for QoS markings for IP/MPLS as traditional MPLS forwarding

All the policies and techniques for implementing QoS that were used in traditional MPLS deployments can be used as-is.

3.6 MPLS Load-Balancing

When multiple equal cost paths to a destination are available, the outgoing path for an incoming packet is selected by using a hashing function in the data plane. Also if a path uses a Layer2 bundle interface (e.g. a LAG using LACP) as the outgoing interface, the outgoing member of the bundle for an incoming packet is selected by computing a hash. With this computed hash value, different traffic flows are load-balanced over the available paths, and all packets of an individual flow are hashed on the same path to avoid out-of-order packets. Which fields of the packet header are used to compute the hash depends on the packet's payload type and the `cef load-balancing fields` configuration in Cisco IOS XR.

To load-balance packets with an IP header, Cisco IOS XR by default uses the 3-tuple hashing. The 3-tuple hashing is also known as Layer-3 or L3 hashing. The 3-tuple hashing function uses the following three elements as input: Source IP address, Destination IP address, and the node's Router-ID. The Router-ID element is included to provide some per node variation to reduce the possibility of polarization. Polarization is the effect of routers in a chain making the same load-balancing decision.[2]

The configuration `cef load-balancing fields L4` changes the default hashing function to 7-tuple, or L4, hashing. The 7-tuple hashing function adds, on top of the elements used for the 3-tuple hashing function, the following fields to the hash computation: Source port (if applicable), Destination port (if applicable), Protocol, and Ingress interface handle.

Packets with an IP payload are best load-balanced by calculating a hash over the IP header fields, even if the packet has an outer MPLS header. If an incoming packet has an outer MPLS header, then the forwarding engine peeks into the packet looking for an IP header. To do this efficiently the forwarding engine only checks the first nibble (4 bits) underneath the bottom MPLS label. If that nibble == 0x4, which is the first nibble in an IPv4 header, then it is assumed that an IPv4 header is present and the hash is calculated using the IPv4 3- or 7-tuple. If that nibble == 0x6, which is the first nibble in an IPv6 header, then it is assumed that an IPv6 header is present and the hash is calculated using the IPv6 3- or 7-tuple. If the nibble has another value then there is no IP header underneath and the load-balancing hash is computed based on the bottom label value and Router-ID.

All packets carried by a regular PseudoWire (PW) should follow the same path through the network to avoid that packets of the pseudowire arrive out-of-order. Different pseudowire traffic flows are best load-balanced by calculating a hash over the bottom label, which is the PW service label. This way, packets with the same PW service label are hashed on the same path. This is automatically accomplished by using the above scheme: if no IP header is found in the packet underneath the label stack, then the hash is calculated on the bottom label and Router-ID.

IETF RFC 6391 introduced a concept of Flow Aware Transport PseudoWires (FAT PW) where the traffic carried within this type of PW consists of a number of distinct flows. It can be desirable to independently load-balance these different flows inside the FAT PW over the available ECMP in the network. This is different from a “regular” PW where all packets of a PW are hashed on the same paths in the network. To load-balance different flows within the FAT PW the PE classifies traffic it sends in the FAT PW, calculates a unique flow label for each flow in the

FAT PW. The PE then pushes this flow label as bottom MPLS label for this flow, underneath the PW service label. The nodes in the network calculate the load-balancing hash on the bottom label, which is the flow label for FAT PW, therefore hashing packets with the same flow label on the same path. This takes care of load-balancing the different flows in the FAT PW over the available ECMP.

A node needs to peek deeper into the header of a packet to load-balance on the IP header fields of the packet underneath the MPLS header. Platforms differ in their capability to peek inside the packet to search for an IP header. If a platform can look N labels deep searching for the bottom of stack label, then this platform calculates the load-balancing hash as indicated in Table 3-2, depending on the label stack and payload type of the incoming packet.

Table 3-2: Load-balancing hash computation

Outer header	**Payload type**	**Hash calculated on**
MPLS stack ≤ N labels	IP Payload	3-tuple or 7-tuple
	Non-IP Payload	Bottom MPLS Label and Router-ID
MPLS stack > N labels	IP Payload	(N+1)th MPLS Label and Router-ID
	Non-IP Payload	(N+1)th MPLS Label and Router-ID

HIGHLIGHT

Segment Routing with MPLS forwarding plane continues to support all the various load-balancing techniques that were available with traditional IP/MPLS forwarding

The introduction of a larger MPLS label stack with certain Segment Routing use-cases and design do not affect the load-balancing features in most router vendors' equipment.

3.7 MPLS MTU Considerations

The Maximum Transmission Unit (MTU) of a protocol layer defines the size of the largest packet ("Protocol Data Unit") that the layer is allowed to transmit over an interface. The default Ethernet Layer-2 MTU in Cisco IOS XR is 1514 bytes. This Layer-2 MTU allows transmission of 1500 bytes Layer-3 packets, including the Layer-3 header. The Layer-2 overhead accounts for the 14 bytes difference. The Layer-2 overhead consists of 6 bytes for destination MAC address, 6 bytes for source MAC address, and 2 bytes for the type or length field. It does not include the preamble, frame delimiter, 4 bytes of Frame Check Sequence (FCS), and inter-frame gap. For a PPP or HDLC frame, the Layer 2 overhead is 4 bytes, making the default interface MTU 1504 bytes for PPP or HDLC interfaces.

The default IP MTU on all interfaces is 1500 bytes. This means that 1500 bytes IP packets, 1500 bytes including the IP header, can be sent on the interface as long as they are not labeled. The default MPLS MTU is 1500 bytes as well, which means that for example labeled IP packets with a single label cannot be longer than 1496 bytes since a single-label MPLS header takes 4 bytes. The ipv4 mtu, ipv6 mtu, and mpls mtu sizes are configured including the L3 header size.

So, in networks using the MPLS data plane, the MTU size of the interfaces should be dimensioned to allow the desired number of MPLS labels on the packet while maintaining the required maximum packet size of the packets transported in MPLS. To maintain the default IP MTU of 1500 bytes, the MPLS MTU must be increased by N*4 bytes, with N the maximum number of labels in the MPLS header.

To increase the MPLS MTU size to allow three labels on top of a 1500 bytes IP packet, the configuration of Example 3-16 can be applied. In the example, the IP MTU is kept at 1500 bytes while the MPLS MTU is increased to 1512 (= 1500 + 3*4) to allow imposition of up to three MPLS labels. Note that to increase the MPLS MTU or IP MTU, the Layer-2 MTU must be increased as well. In the example the Layer-2 MTU is configured as 1526 (= 1512 + 14).

Example 3-16: Configuring and verifying non-default MTU in IOS XR

```
RP/0/RSP0/CPU0:R1#show running-config interface TenGigE0/1/0/1
interface TenGigE0/1/0/1
 description R1toR2
 cdp
 mtu 1526
 ipv4 mtu 1500
 ipv4 address 99.1.2.1 255.255.255.0
 ipv6 mtu 1500
 ipv6 address 2001:db8::99:1:2:1/112
!

RP/0/RSP0/CPU0:R1#  show interfaces TenGigE0/1/0/1 | i MTU
  MTU 1526 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
RP/0/RSP0/CPU0:R1#  show im database interface TenGigE0/1/0/1

View: OWN - Owner, L3P - Local 3rd Party, G3P - Global 3rd
Party, LDP - Local Data Plane
      GDP - Global Data Plane, RED - Redundancy, UL - UL

Node 0/1/CPU0 (0x831)

Interface TenGigE0/1/0/1, ifh 0x06000100 (up, 1526)
  Interface flags:          0x000000000110059f
(ROOT_IS_HW|IFCONNECTOR
                            |IFINDEX|SUP_NAMED_SUB|BROADCAST|CONF

                            |DATA|CONTROL)


  Encapsulation:
ether
```

```
  Interface type:          IFT_TENGETHERNET

  Control parent:
None
  Data parent:
None
  Views:
GDP|LDP|L3P|OWN

  Protocol        Caps (state, mtu)
  --------        -----------------
  None            ether (up, 1526)
  arp             arp (up, 1512)
  clns            clns (up, 1512)
  ipv4            ipv4 (up, 1500)
  mpls            mpls (up, 1512)
  ipv6            ipv6_preswitch (up, 1512)
  ipv6            ipv6 (up, 1500)
  ether_sock      ether_sock (up, 1512)

RP/0/RSP0/CPU0:R1#  show ipv4 interface TenGigE0/1/0/1 | i MTU
  MTU is 1526 (1500 is available to IP)
RP/0/RSP0/CPU0:R1#  show ipv6 interface TenGigE0/1/0/1 | i MTU
  MTU is 1526 (1500 is available to IPv6)
RP/0/RSP0/CPU0:R1#  show mpls interfaces TenGigE0/1/0/1 private
location 0/1/C$
Interface      IFH        MTU
-------------- ---------- -----
Te0/1/0/1      0x06000100  1512
```

End hosts typically still use an IP MTU of 1500 bytes. Networks present the standard default IP MTU of 1500 bytes to end hosts, while internal links have a higher MTU size.

In most networks today, however, where high capacity fiber links are widely used, it is not uncommon to see MTU configurations taking into account jumbo frames up to 9216. The arithmetic to allow for the necessary MPLS label stack though still remains the same as does the best practices to use techniques like path MTU discovery mechanisms. In such modern networks, the Segment Routing MPLS label stack does not really pose as an overhead for most types of traffic.

When considering Segment Routing enabled mechanisms like TI-LFA (described later in chapter 9) or Traffic Engineering (which will be covered in the next part of this book), the MPLS label stack on the packet could potentially grow as the Segment List size increases. As illustrated later in the TI-LFA chapter, in most networks a small number of SIDs are sufficient to provide the necessary protection. Also, as we will see later with Segment Routing Traffic Engineering, there are techniques that allow for compression of the number of SIDs required. Contrary to what one might fear, for most practical purposes the MPLS label stack on a packet does not grow very big. It is, however, required that the necessary headroom for the maximum label stack anticipated be made when setting up the MTUs across the network.

HIGHLIGHT

When using Segment Routing features, it is a good practice to account for the maximum label stack depth expected when provisioning MTU across the network

"Leveraging the existing MPLS forwarding architecture by SR has significantly eased its adoption and deployment by a significant number of operators today within a relatively short span of time of its development. Most vendors' current routing platforms only need a software upgrade for supporting the basic SR feature set."

— Ketan Talaulikar

3.8 Summary

- The traditional MPLS data plane is reused without any modification for Segment Routing
- Label operations, PHP behavior, TTL processing, EXP/TC processing, load-balancing techniques and MTU handling are used similarly for Segment Routing
- Segment Routing MPLS forwarding entries on routers in the network are much simpler to follow and troubleshoot with Global Prefix SID labels when using homogenous SRGB

3.9 References

- [MTU] "MTU Behavior on Cisco IOS XR and Cisco IOS Routers", http://www.cisco.com/c/en/us/support/docs/ios-nx-os-software/ios-xr-software/116350-trouble-ios-xr-mtu-00.html
- [RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, https://datatracker.ietf.org/doc/rfc2474.
- [RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, DOI 10.17487/RFC2475, December 1998, https://datatracker.ietf.org/doc/rfc2475.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, https://datatracker.ietf.org/doc/rfc3031.
- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, https://datatracker.ietf.org/doc/rfc3032.
- [RFC3270] Le Faucheur, F., Wu, L., Davie, B., Davari, S., Vaananen, P., Krishnan, R., Cheval, P., and J. Heinanen, "Multi-Protocol Label Switching (MPLS) Support of Differentiated Services", RFC 3270, DOI 10.17487/RFC3270, May 2002, https://datatracker.ietf.org/doc/rfc3270.
- [RFC4950] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "ICMP Extensions for Multiprotocol Label Switching", RFC 4950, DOI

10.17487/RFC4950, August 2007, https://datatracker.ietf.org/doc/rfc4950.

- [RFC5462] Andersson, L. and R. Asati, “Multiprotocol Label Switching (MPLS) Label Stack Entry: “EXP” Field Renamed to “Traffic Class” Field”, RFC 5462, DOI 10.17487/RFC5462, February 2009, https://datatracker.ietf.org/doc/rfc5462.
- [RFC6391] Bryant, S., Ed., Filsfils, C., Drafz, U., Kompella, V., Regan, J., and S. Amante, “Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network”, RFC 6391, DOI 10.17487/RFC6391, November 2011, https://datatracker.ietf.org/doc/rfc6391.

[1] The name of the fields is changed to “Traffic Class” by IETF RFC 5462

[2] “ASR9000/XR: Load-balancing architecture and characteristics”, https://supportforums.cisco.com/document/111291/asr9000xr-loadbalancing-architecture-and-characteristics

4 MANAGEMENT OF THE SRGB

Segment Routing Global Block (SRGB) provisioning is used to carve out a label for Global Segments when using MPLS data plane for Segment Routing. This needs to be done on each Segment Routing aware router in the network. Previously, in chapter 3, “Segment Routing MPLS Data Plane”, we have introduced the SRGB concepts and how it could be provisioned and used in most of the deployments – i.e. a single common range being used across the network (homogenous SRGB). In this chapter, we will look more closely at more aspects related to the management of SRGB.

4.1 SRGB Size

The size of SRGB dictates the number of Global Segments that can be used and in the most typical case, this would map to the number of Prefix Segments that need to be handled on the router. This in turn relates, in most typical cases, to the number of routers in the IGP network (as discussed previously in chapter 2, “Fundamentals of Segment Routing”, we need at least one Node Segment per router) and additionally other Prefix Segments that are required e.g. for other service loopback addresses on some routers, for usage like Anycast Prefix Segments or for prefixes which are being redistributed from other (parts of) network. A clearer understanding of required scale of Prefix Segments will come as we go through further chapters in this book.

When deploying Segment Routing, the current and future network growth and design are important to be kept in mind when determining the SRGB size required. It is possible to extend/grow the SRGB size and/or add more ranges as we will see further in this chapter. However, upfront planning when introducing Segment Routing can ensure stable and consistent SRGB which in turn can simplify the network operations greatly.

While the SRGB label values can be different per node, the size of the SRGB range (i.e. the amount of labels in the range), should ideally be the same on all nodes in the SR Domain. The size of the SRGB determines the highest possible SID index that can be used for a Global Segment. For example a SID index of 2000 is not valid if the SRGB only counts 1000 labels. If different sizes are used on different nodes in the SR Domain, then the smallest range size of all SRGBs specifies the range size that can be used in the SR Domain. Using Prefix SID index(es) inconsistent with

SRGB range sizes in routers across the SR domain could result in traffic getting dropped or mis-routed as a router may not have a valid global label that it can use for forwarding to its neighbor that has a smaller SRGB range.

It is perfectly possible though to grow or increase the SRGB size in the network in a rolling manner across routers and then to start using the increased index space once the update is done throughout the network.

In Cisco IOS XR, the default SRGB is the label range from 16000 to 23999 that provides for 8000 Segments which are sufficient for most deployments. For very large scale networks there are some design approaches suggested in chapter 10, "Large Scale Interconnect with Segment Routing" which can be done using this same default SRGB size. A non-default SRGB can be freely configured within the range 16000 to 2^{20}-1 (~1 Million) for Cisco IOS XR or up to the platform limit if that is lower than 1 Million. The maximum SRGB size that can be advertised in IGP is 2^{24}, which is larger than the entire MPLS label range. Currently, the maximum configurable range size of the SRGB in Cisco IOS XR is 64k labels in general and up to the full MPLS label ranges on some platforms.

HIGHLIGHT

SR is not limited to 1 million global prefix segments.

Hyper-scale SR-based DC designs involving 10's of millions of addressable endpoints are described in IETF draft-filsfils-spring-large-scale-interconnect.

The same approach has been used to design large-scale aggregation networks of 100's of thousands addressable endpoints while using an SRGB of a few thousands of entries

These designs will be illustrated in chapter 10, “Large Scale Interconnect with Segment Routing”.

4.2 Multi-Range SRGB

The computation of the local label value of a Prefix-SID is straightforward if the SRGB consists of a single range of labels. In such case the local label is simply computed by adding the SID index to the SRGB Base value.

The Segment Routing specification permits using an SRGB that consists of multiple disjoint label ranges. If multiple disjoint label ranges make up the SRGB on a node, then the label ranges must be concatenated and the SID index is then used as an offset in that concatenated range.

For example, a node uses the following label ranges as SRGB: [800000-801999], [700000-701999], [900000-903999]. Note that the order of the ranges must stay constant. To find the local label value of a Global Segment on this node, the three label ranges are concatenated and the SID index is then used as an offset in that concatenated range. The SID index to label value mapping is shown in Table 4-1. The second column of this table contains the label values of the concatenated three label ranges (only first and last label value of each label range is shown). The first column contains the SID index, a series of incrementing numbers starting at 0. For example SID index 1999 has local label value 801999 and SID index 2500 has label value 700500.

Table 4-1: Multiple label range SRGB example

SID index	SID label value
0	800000
…	…
1999	801999

2000	700000
…	…
3999	701999
4000	900000
…	…
7999	903999

The provisioning of SRGB with multiple ranges is to allow for growing the SRGB and/or to allow selection of label space differently for different router platforms. However, doing so could make the management of SRGB complex across the network. Currently this functionality is not supported in the Cisco IOS XR implementation. This book assumes that a single label range is used in the SRGB, which is the recommended deployment model.

4.3 SRGB Label Range

All Cisco IOS XR platforms that support Segment Routing can accommodate reservation of the SRGB anywhere between 16,000 and $2^{20} - 1$. The default SRGB for Cisco IOS XR platforms is the range of labels from 16000 to 23999. Other vendors also adopted this label range for the SRGB by default. A non-default SRGB can be configured if desired.

If the limitations of platforms deployed in a network force the use of different SRGBs on different nodes, then the number of different SRGBs in the network should be limited. If possible it is recommended to select SRGB base values that can easily be mapped to each other by the human operator (e.g. SRGB base is a multiple of 10000).

When using the same SRGB on all nodes, then Global Segments are still distributed as globally unique SID indexes. But these SID indexes map to the same local label value on each node in the SR Domain. Therefore: same local label value on all nodes = global label value.

"Using the same SRGB on all nodes within the SR Domain has undoubtedly significant advantages in simplification of administration, operation and troubleshooting. Also programming the network is simplified by using this model, and Anycast Segments can be used without added complexity. Using the same SRGB network-wide is expected to be a deployment guideline. As mentioned in IETF draft-filsfils-spring-segment-routing-use-cases-01: "Several operators have indicated that they would deploy the SR technology in this way: with a single consistent SRGB across all the nodes. They motivated their choice based on operational simplicity (…)"."

— *Kris Michielsen*

Figure 4-1 shows a simple topology: a chain of six nodes. The table under each node represents the label space where the SRGB can be located.

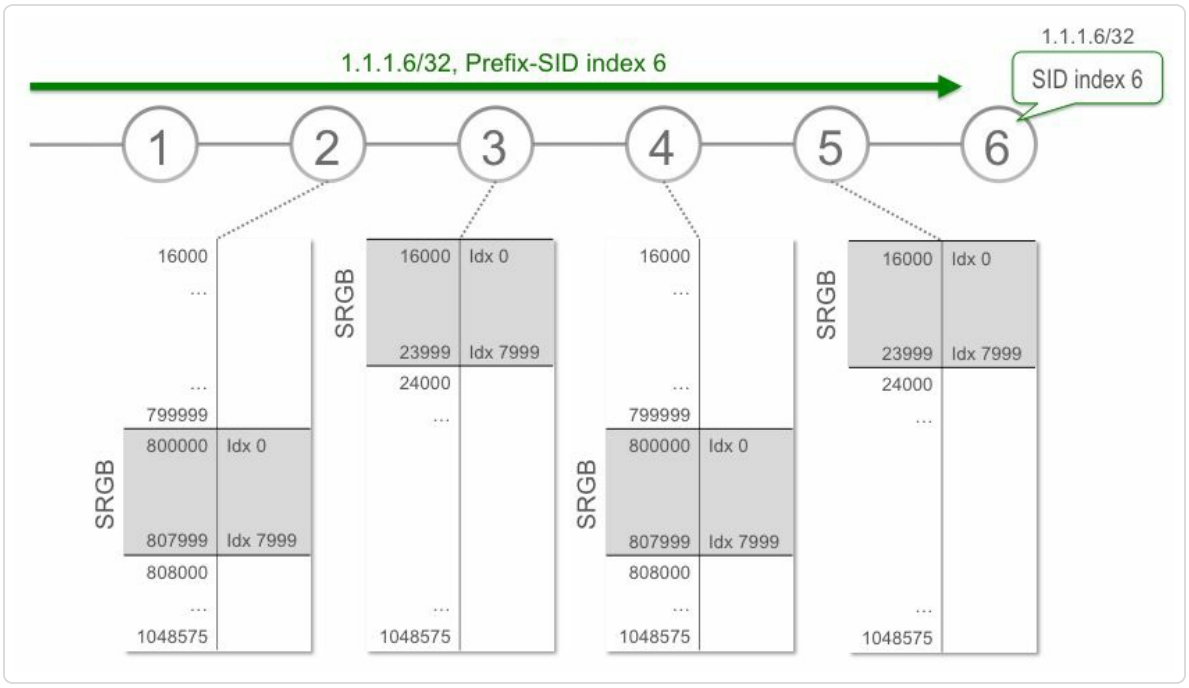


Figure 4-1: Not recommended, but possible SRGB allocation (1)

The operator configures Segment Routing on all nodes in this topology. He decides to use the default SRGB [16000-23999] on Node1, Node3, Node5 and Node6.

Although not recommended, the operator uses a different SRGB on Node2 and Node4, starting from 800000 and with a range size of 8000. Possibly Node2 and Node4 are another vendor's platform that only supports an SRGB starting at 800000. However even so, following the recommended deployment guideline, all nodes could then use an SRGB starting at 800000. This is possible for IOS XR devices.

Node6 advertises a prefix 1.1.1.6/32 with a prefix-SID index of 6. All six nodes each install their local label for Prefix-SID index 6 in their forwarding table. See Figure 4-2.

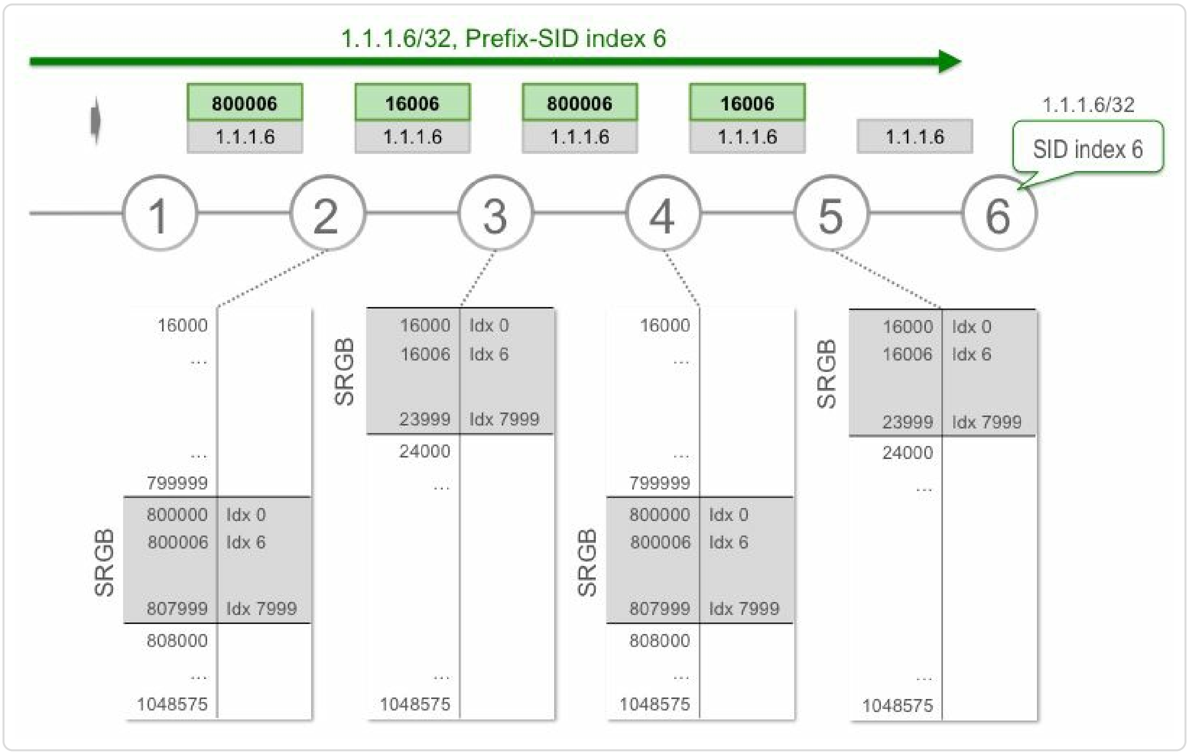


Figure 4-2: Not recommended, but possible SRGB allocation (2)

Node1, Node3, Node5 and Node6, the nodes that use the default SRGB, install a local label 16006 for this Prefix Segment. This is the first label in the SRGB, 16000, plus the Prefix-SID index 6. Node2 and Node4, the nodes that uses the SRGB starting at 800000, install a local label 800006 for this Prefix Segment (= 800000 + 6).

The outgoing label that each node installs in the forwarding table for this Prefix Segment depends on the SRGB of the downstream node, the nexthop node on the shortest path to the destination Node6. The outgoing label is the local label that the downstream neighbor allocated for the Prefix-SID. Node2 advertises a SRGB [800000-807999], thus Node2 allocates the local label 800006 for the Prefix Segment with SID index 6 (800000 + 6 = 800006). Therefore Node1 installs an outgoing label 800006 for this Prefix Segment. The downstream neighbor of Node2 for that Prefix Segment is Node3. Node3 advertises SRGB [16000-23999], therefore Node2 installs outgoing label 16006 (=16000 + 6) for that Prefix Segment. And so on for Node3, Node4, and Node5.

A packet arrives at Node1 with top label 16006, the label for the Prefix Segment of 1.1.1.6/32. Node1 swaps label 16006 with 800006, the outgoing label towards Node2 for that Prefix Segment, and forwards to Node2. Node2 swaps the label 800006 to 16006 again, which is the outgoing label towards Node3 for this Prefix Segment, and forwards to Node3. Node3 swaps the label to 800006 and Node4 swaps to 16006 again. Node5 pops the label and forwards it to destination Node6.

The output of a traceroute from Node1 to Node6 in Example 4-1 shows how the label changes on its path to the destination.

Example 4-1: Traceroute in multi-SRGB network

```
RP/0/0/CPU0:xrvr-1#traceroute 1.1.1.6

Type escape sequence to abort.
Tracing the route to 1.1.1.6

 1  99.1.2.2 [MPLS: Label 800006 Exp 0] 19 msec  19 msec  9
msec
 2  99.2.3.3 [MPLS: Label 16006 Exp 0] 9 msec  9 msec  9 msec
 3  99.3.4.4 [MPLS: Label 800006 Exp 0] 9 msec  9 msec  19 msec
 4  99.4.5.5 [MPLS: Label 16006 Exp 0] 9 msec  9 msec  19 msec
 5  99.5.6.6 9 msec  9 msec  9 msec
```

Using different SRGBs on different nodes in the network forces the operator to always consider the SRGB of each node while troubleshooting a problem. In this example, only two different SRGBs are used on nodes in the network. Having multiple different SRGBs in the network makes it even less straightforward. It would become similar to tracing an LDP LSP through the network, except that the Prefix-SID labels are not totally random, contrary to LDP labels.

Note that the complexity increases in multi-label label stacks: each of the labels in the stack needs to match the correct node's SRGB. This is illustrated in Figure 4-3.

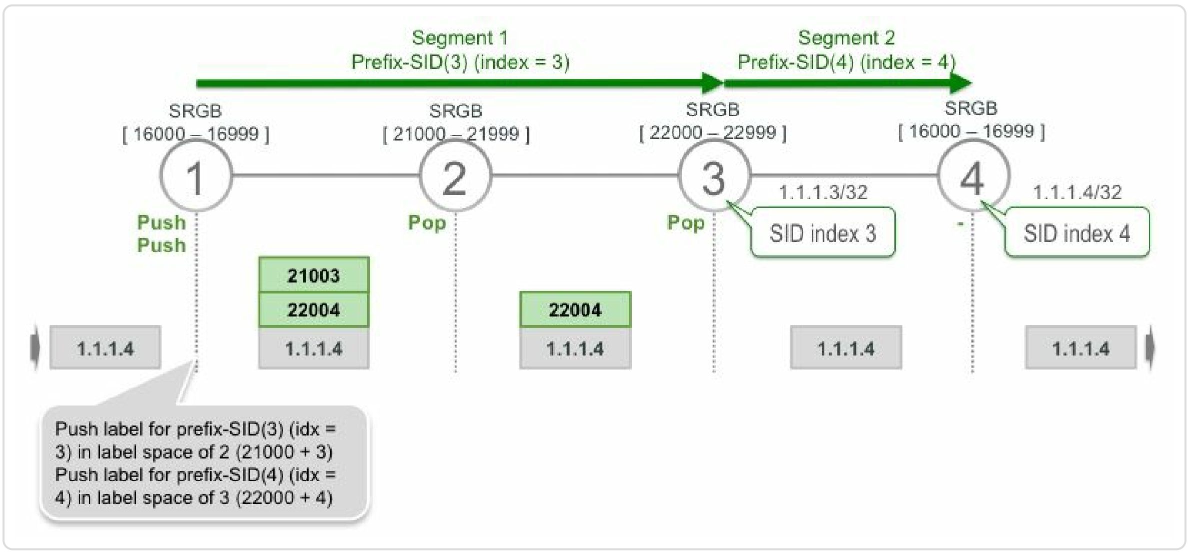


Figure 4-3: Multi-label stacks and different SRGBs

To steer traffic towards Node4 via Node3, Node1 uses two segments: {Prefix-SID(3), Prefix-SID(4)}. Node1 needs to consider the SRGB of the right node to derive which label values to push on the packets.

To compute the label value for Prefix-SID(3), Node1 uses the SRGB of its nexthop towards Node3, which is Node2. Node1 computes the label for Prefix-SID(3) by adding the SID index 3 to the SRGB base of Node2: 21000 + 3 = 21003.

For Prefix-SID(4), Node1 uses the SRGB of Node3 to compute the label value. Node3 receives the packets with the second Prefix-SID label on top.

Node1 needs to compute the second label value in the label context of Node3.

Node1 imposes the label stack {21003, 22004} on the packets.

> "Following the best practice deployment guideline to use the same SRGB on all nodes simplifies operating and troubleshooting the network. In that case, the packet is transported with the same predictable label value, the global label value for this Prefix Segment, all the way to the destination. Simple, predictable and much easier to troubleshoot."
>
> — *Kris Michielsen*

Figure 4-4 shows the same topology as in Figure 4-1, but this time all nodes are using the same SRGB [16000-23999].

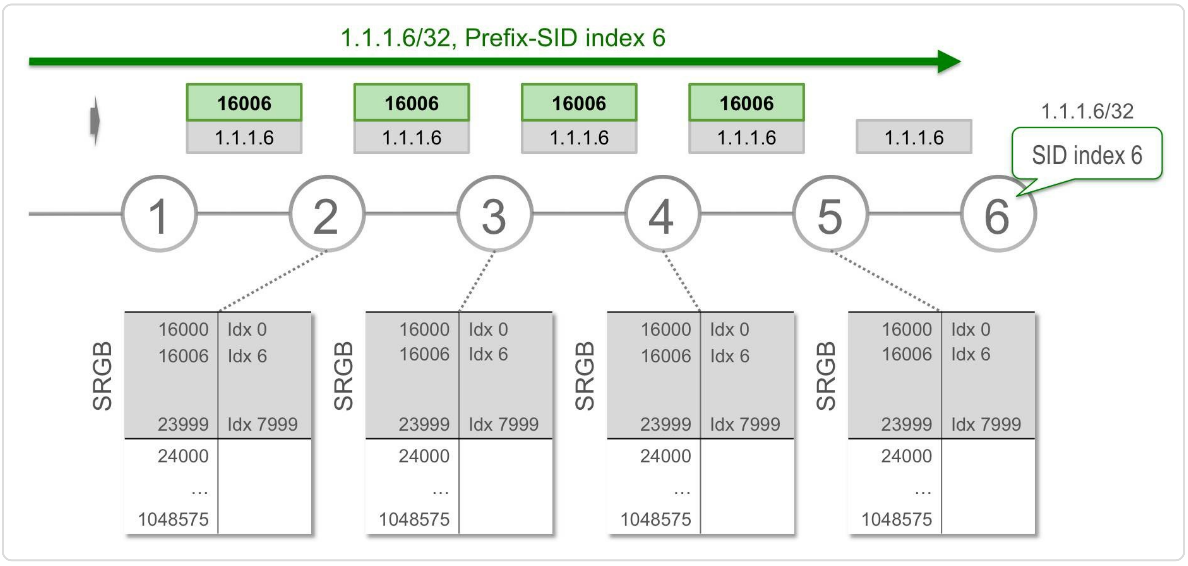


Figure 4-4: Recommended SRGB allocation

A packet arrives at Node1 with top label 16006, the label for the Prefix Segment of 1.1.1.6/32. Node1 swaps label 16006 with 16006, the outgoing label towards Node2 for that Prefix Segment, and forwards to Node2. The following nodes again swap 16006 with 16006 until Node5 pops the label and forwards it to destination Node6.

The output of a traceroute from Node1 to Node6 in Example 4-2 shows how the same label is used on the entire to the destination.

Example 4-2: Traceroute in same-SRGB network

```
RP/0/0/CPU0:xrvr-1#traceroute 1.1.1.6

Type escape sequence to abort.
Tracing the route to 1.1.1.6

 1  99.1.2.2 [MPLS: Label 16006 Exp 0] 19 msec  19 msec  9 msec
 2  99.2.3.3 [MPLS: Label 16006 Exp 0] 9 msec  9 msec  9 msec
 3  99.3.4.4 [MPLS: Label 16006 Exp 0] 9 msec  9 msec  19 msec
 4  99.4.5.5 [MPLS: Label 16006 Exp 0] 9 msec  19 msec  9 msec
 5  99.5.6.6 9 msec  9 msec  9 msec
```

No need to think about the SRGB of each node, since each node uses the same one. An operator can directly verify the forwarding entry for this Prefix Segment on each node without having to verify which node advertises what. Forwarding entries are easily verified; errors jump out and are easily detected.

4.4 SRGB and Anycast Segments

When using Anycast Segments in a MPLS Segment Routing Domain, all nodes in the same anycast set should use the same SRGB for simplified operations. Figure 4-5 illustrates the problem that occurs when not using a common SRGB.

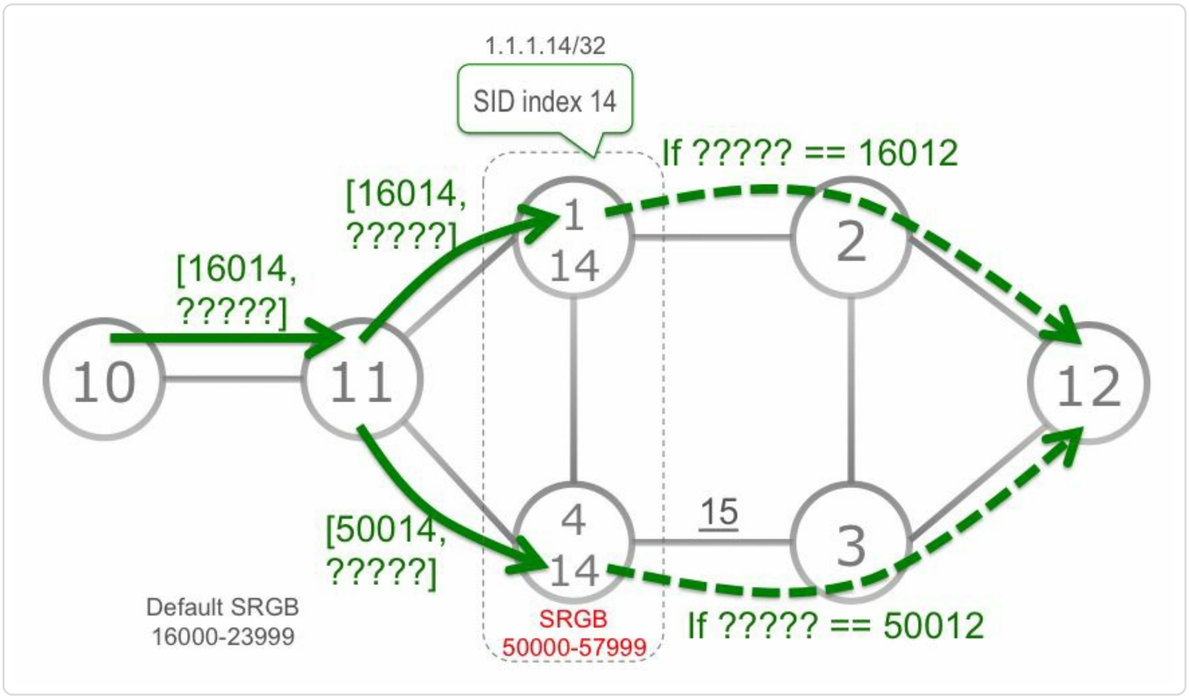


Figure 4-5: Anycast-SID problem when not using same SRGB

In this topology, Node1 and Node4 are part of an anycast set. Besides their own individual Node-SID with a SID index equal to the number of the node, both nodes also advertise the anycast prefix 1.1.1.14/32 with an associated Anycast-SID index 14. Node10 steers packets to Node12 via one of the nodes Node1 or Node4, using the Segment List {Anycast-SID(14), Prefix-SID(12)}. Therefore Node10 pushes 2 labels on the packets. The first label is the Anycast-SID label of the anycast set {Node1, Node4}. The second label is the Prefix-SID label of Node12.

Node10 can easily derive the label for the Anycast-SID: it is the label 16014 allocated by Node11 for the Anycast-SID index 14 (16000 + 14 = 16014).

But what is the next label that Node10 needs to push on the label stack for Prefix-SID(12)? If the packet would go via Node1, then the second label would have to be 16012 since Node1 uses the default SRGB [16000-23999] (16000 + 12 = 16012). If the packet would go via Node4, then the second label would have to be 50012 since Node4 uses SRGB [50000-57999] (50000 + 12 = 50012).

It turns out that there is no single label value that Node10 can push on the packets to steer them via the anycast set {Node1, Node4} to the destination Node12 if the nodes in the anycast set do not use the same SRGB.

Figure 4-6 illustrates the case where all nodes in the anycast set use the same SRGB. For illustration purposes Node1 and Node4 use the same SRGB [50000-57999], but different from the other nodes' SRGB. Best is to use the same SRGB on all nodes in the network.

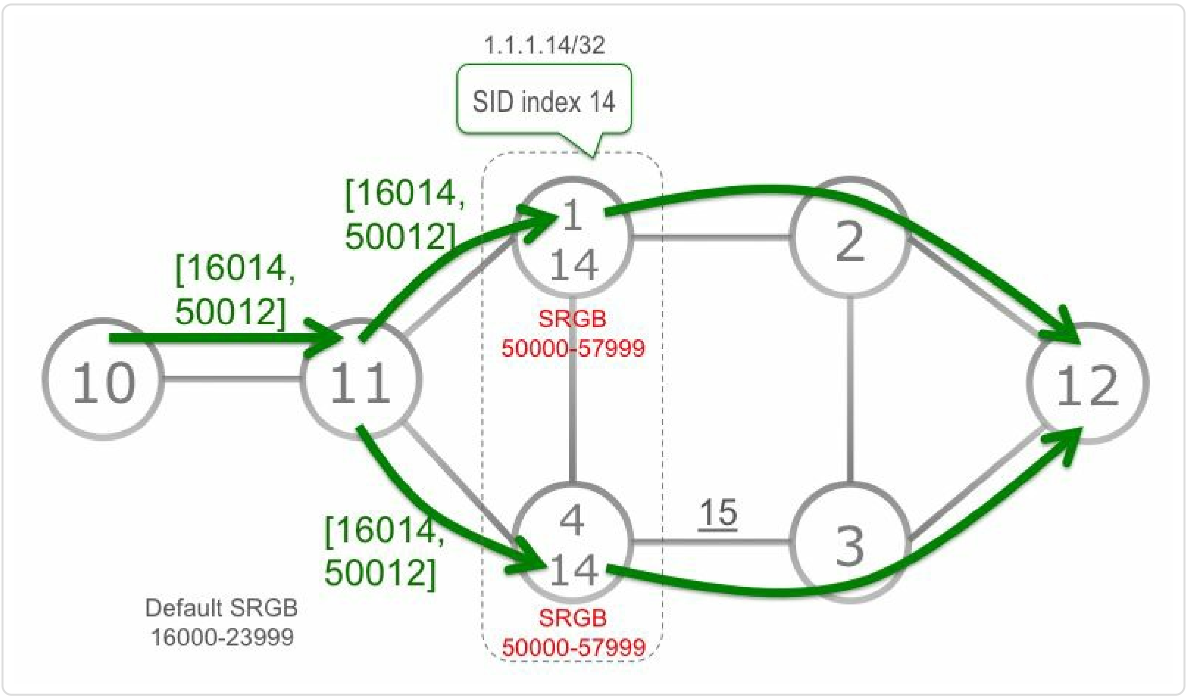


Figure 4-6: Anycast-SID requires same SRGB

Node10 can now also easily determine the second label on the stack. Both Node1 and Node4 allocate the same local label for Prefix-SID index 12: 50012 (= 50000 + 12).

The conclusion is that the simple and most desirable solution is to use the same SRGB on all nodes in the anycast set. Another, more complicated solution for using Anycast-SIDs with different SRGBs is described in the IETF document draft-psarkar-spring-mpls-anycast-segments. Cisco IOS-XR implementations only support Anycast Segment operations currently with a homogenous SRGB.

"The implementation and operation for Anycast-SID becomes simple and straightforward when using a homogenous SRGB across the network. While there are proposals for supporting Anycast-SID with non-homogenous SRGB, it is evident very quickly for anyone that reviews those proposals how sometimes following the prescribed design guideline is much simpler. As mentioned in the opening chapter of this book, following the prescribed design guidelines can lead to simpler operations in the network.”

— Ketan Talaulikar

4.5 SRGB Configuration

Although the Segment Routing Architecture does not mandate it, the SRGB can be seen as a property of a node. The SRGB indicates which labels are to be used for Global Segments on a node, independent of the Segment Routing protocol. In Cisco IOS XR the SRGB is currently not entirely protocol independent. The SRGB can be globally configured and all Segment Routing protocols use that SRGB by default. But IS-IS and OSPF can also use an IGP instance-specific SRGB, configured under the routing protocol configuration, and for these protocols the global SRGB configuration is optional. Multiple IGP instances and BGP can use the same SRGB or use different but non-overlapping SRGBs in Cisco IOS XR.

Note that by only configuring a global SRGB in Cisco IOS XR the SRGB is not actually allocated. A protocol that uses the SRGB must be configured and enabled with Segment Routing as it is that protocol (or those protocols) that reserves the SRGB label range from the router's label space.

Modifying the SRGB configuration is disruptive for traffic. You need to carefully plan any SRGB changes on a router in production. If the SRGB is changed or a non-default SRGB is configured while the router is operational, a reload is generally required since labels within the new SRGB may already be in use by other MPLS applications. In that case the new SRGB cannot be allocated, as there is currently no mechanism in place to gracefully release labels that are in use.

When modifying an SRGB, the old SRGB is released first. If the new SRGB label range is not entirely available then Segment Routing forwarding is dysfunctional until the SRGB can be allocated. A reload is generally required in that case. The syslog message in Example 4-3 is logged when Segment Routing was active and the SRGB configuration was changed.

Example 4-3: Syslog message for changing SRGB

```
RP/0/0/CPU0:Feb 17 13:08:45.628 : isis[1011]: %ROUTING-ISIS-6-
SRGB_INFO : SRGB info: 'Segment routing temporarily disabled on
all topologies and address families because the global block is
being modified'
```

An allocated SRGB can be verified in the `show mpls label table detail` output, as shown in Example 4-4. In this example ISIS allocated the default SRGB that starts at label 16000 and has a size of 8000. Without the `detail` keyword only the first label in the range is displayed without the range size. The output also shows which client owns the label block, `ISIS(A): 1` in this example.

Example 4-4: SRGB allocation single protocol

```
RP/0/0/CPU0:xrvr-4#show mpls label table label 16000 detail
Table Label    Owner                              State  Rewrite
----- -------- ---------------------------------- ------ -------
0     16000    ISIS(A):1                          InUse  No
  (Lbl-blk SRGB, vers:0, (start_label=16000, size=8000)
```

Example 4-5 illustrates the case where multiple IGP instances and BGP are using the same SRGB. In this example `SIS(A):1`, `ISIS(A):2`, `OSPF(A):ospf-1` and `BGP-VPNv4(A):bgp-default` all use the same SRGB [16000-23999].

Example 4-5: srgb allocation multiple protocols

```
RP/0/0/CPU0:xrvr-4#sh mpls label table label 16000 det
Table Label   Owner                           State  Rewrite
----- ------- ------------------------------- ------ -------
0     16000   ISIS(A):1                       InUse  No
              BGP-VPNv4(A):bgp-default        InUse  No
              OSPF(A):ospf-1                  InUse  No
              ISIS(A):2                       InUse  No
  (Lbl-blk SRGB, vers:0, (start_label=16000, size=8000)
```

The SRGB configuration for the IGPs in Cisco IOS XR uses a hierarchical model. The SRGB for the IGPs can be left as the default, globally configured or configured per-IGP instance. The preference order is (from most preferred to least preferred): (1) per-IGP configuration, (2) global configuration, (3) default.

BGP only uses the globally configured SRGB. This global SRGB configuration has no default. If the SRGB range [16000-23999] is desired for BGP, then it needs to be explicitly configured globally as `segment-routing global-block 16000 23999`. An example of this configuration is shown in lines 15 and 16 of Example 4-6.

Table 4-2 shows the resulting SRGB allocation for different configuration combinations. The columns indicate different global SRGB configurations; the rows indicate different per-IGP configurations.

Table 4-2: Cisco IOS XR SRGB configurations

Global SRGB → **IGP instance SRGB ↓**	**None**	**SRGB1**	**SRGB2**
None	IGP: default BGP: none	IGP: SRGB1 BGP: SRGB1	IGP: SRGB2 BGP: SRGB2
SRGB1	IGP: SRGB1 BGP: none	IGP: SRGB1 BGP: SRGB1	IGP: SRGB1 BGP: SRGB2

The first column is for the case without global SRGB configuration, the second column for a global SRGB1, and the third column for a global SRGB2. The rows indicate different per-IGP SRGB configurations. The first row for the case without per-IGP SRGB configuration, the second row for a per-IGP SRGB1.

Example 4-6 illustrates an atypical SRGB configuration in Cisco IOS XR. The general recommendation is to use the same SRGB for all protocols on a node. The example illustrates that other configurations are possible. ISIS has a non-default per-IGP SRGB configuration that is different from the global SRGB configuration that is used by OSPF and BGP.

Example 4-6: multi-SRGB configuration example

```
router isis 1
 address-family ipv4 unicast
  segment-routing mpls
!
router isis 2
 segment-routing global-block 700000 709999
 address-family ipv4 unicast
  segment-routing mpls
!
router ospf 1
 segment-routing mpls
!
router bgp 1
!
segment-routing
 global-block 16000 23999
```

The configuration of Example 4-6 results in the SRGBs allocation as shown in Example 4-7.

Example 4-7: multi-SRGB allocation example

```
RP/0/0/CPU0:xrvr-4#sh mpls label table detail | e "SR
Adj|InUse  Yes"
Table Label   Owner                           State  Rewrite
----- ------- ------------------------------- ------ -------
0      16000  ISIS(A):1                       InUse  No
              BGP-VPNv4(A):bgp-default        InUse  No
              OSPF(A):ospf-1                  InUse  No
  (Lbl-blk SRGB, vers:0, (start_label=16000, size=8000)
0     700000  ISIS(A):2                       InUse  No
  (Lbl-blk SRGB, vers:0, (start_label=700000, size=10000)
```

ISIS 2 uses the ISIS-specific SRGB [700000-709999] while ISIS 1, OSPF, and BGP use the global SRGB [16000-23999].

HIGHLIGHT

Change in SRGB configuration would be disruptive for services; likely a reload is required when changing SRGB to a non-default value

It is recommended to use specify SRGB with the global configuration in Cisco IOS-XR in most cases so that all IGP instance(s) and BGP use the same.

In specific cases, it is possible to configure SRGB under individual IGP instances in Cisco IOS-XR e.g. in cases where the network is partitioned using multiple IGP instances.

The configuration of SRGB is something that is very much vendor and platform specific aspect. One should study this aspect carefully in the routing platforms where Segment Routing is going to be enabled.

4.5.1 Cisco IOS XR SRGB Implementation

The Label Switch Database (LSD) in Cisco IOS XR is a central repository of label switching information. Managing the allocation of local labels is one of the tasks handled by LSD. In order to request allocation of local labels, MPLS applications such as IGP, LDP, RSVP, MPLS static, and so on, have to register as a client with LSD. Via this client connection, the MPLS application can request local labels from LSD.

In the software versions that are Segment Routing capable, the label space is by default carved as follows:

- The range 0 to 15 is reserved for special-purpose labels
- The range 16 to 15999 is reserved for static MPLS labels. This range is used by MPLS applications that require static labels, such as MPLS-TP, MPLS static, and Pseudowire.
- The range 16000 to 23999 is preserved for use as the default Segment Routing Global Block.
- The range 24000 to the maximum is used for dynamic label allocation. Maximum is 2^{20}-1 or up to the platform limit.

Most MPLS applications use local labels that are dynamically allocated by LSD. They just ask *a* label from LSD and LSD gives them a local label that it picked from the dynamic label range. Examples of MPLS applications that use dynamic labels are LDP, RSVP, L2VPN, BGP, TE, IS-IS and OSPF. By default, LSD allocates local labels from the label range starting at 24000 and outside of any allocated non-default SRGB label range.

LSD uses a specific mechanism to allocate Segment Routing Global Block label ranges at boot time. It provides an opportunity for its clients which are Segment Routing control plane protocols (i.e. ISIS, OSPF and BGP) to first register with it and perform their configured SRGB allocations. If Segment Routing is not enabled, there is no allocations to be made and in this case LSD only tries to preserve the default SRGB range. If Segment Routing is enabled, then LSD will reserve the SRGB ranges as asked by the control plane protocols. Only after the SRGB allocations are handled, does LSD open up to handling local label allocations from all other MPLS applications.

The software also provides the possibility to enable Segment Routing at some distant point in time, when the system has already been in use for some time, without getting into any contention with other MPLS applications. This is because LSD has, at boot time, already preserved the default SRGB label range, from 16,000 to 23,999, for Segment Routing. This is the case for any Segment Routing capable Cisco IOS XR software release, even if Segment Routing is not (yet) enabled.

Example 4-8 shows how the current dynamic label range can be verified using the `show mpls label range` command.

Example 4-8: MPLS dynamic label range

```
RP/0/RP0/CPU0:R1#show mpls label range
Range for dynamic labels: Min/Max: 24000/1048575
```

If the preservation of this default SRGB label range is not required, the dynamic label range can be expanded to include the labels in the range [16000-23999] by using the `mpls label range <min> <max>` configuration. This configuration should only be used when using, or planning to use, a non-default SRGB label range. See Example 4-9.

Example 4-9: modify MPLS dynamic label range

```
RP/0/0/CPU0:iosxrv-1#show mpls label range
Range for dynamic labels: Min/Max: 24000/1048575
RP/0/0/CPU0:iosxrv-1#config
```

```
RP/0/0/CPU0:iosxrv-1(config)#mpls label range 16000 1048575
RP/0/0/CPU0:iosxrv-1(config)#commit
RP/0/0/CPU0:iosxrv-1(config)#end
RP/0/0/CPU0:iosxrv-1#show mpls label range
Range for dynamic labels: Min/Max: 16000/1048575
```

When enabling Segment Routing on a system, the Segment Routing control plane requests its SRGB label range from LSD. If other MPLS applications are active on the system then one or more labels in the requested SRGB label range may already be allocated to these other MPLS applications. If not all labels in the requested range are available, then the SRGB allocation fails and Segment Routing forwarding entries will not be installed. Currently there is no mechanism in place to release the labels allocated by MPLS applications in a non-disruptive manner and a system reboot would be required in that case to activate Segment Routing forwarding. However, if none of the labels in the requested SRGB label range is in use, then the SRGB can immediately be allocated, Segment Routing forwarding entries can immediately be installed and Segment Routing can immediately be used without requiring a system reboot. Note that such a reboot would not be required when using the default SRGB since that would be always available to reserve/allocate at any point.

Note that some platforms (for example ASR9000 series) require configuration of the MPLS label range in order to expand the dynamic label range to the full MPLS label space. If a MPLS label range configuration is in place and includes (part of) the SRGB label range, then the SRGB label range is *not* preserved. In that case future activation of Segment Routing may require a system reboot.

HIGHLIGHT

Enabling Segment Routing by using the default SRGB on a Cisco IOS-XR system can be done in production without requiring a disruptive reboot operation.

Figure 4-7 illustrates the allocation of the default SRGB at boot time.

After the router image booted, LSD has carved the label range in the sections shows in Figure 4-7:

- Labels from 0 to 15999 used for special-purpose and static labels
- Labels from 16000 to 23999 preserved for SRGB
- Label range from 24000 to 1 million, available for dynamic labels

LSD first waits for its Segment Routing protocol clients. IS-IS registers as a client with LSD and IS-IS is configured to use the default SRGB. Hence IS-IS requests the default SRGB local label range, 16000 to 23999, from LSD.

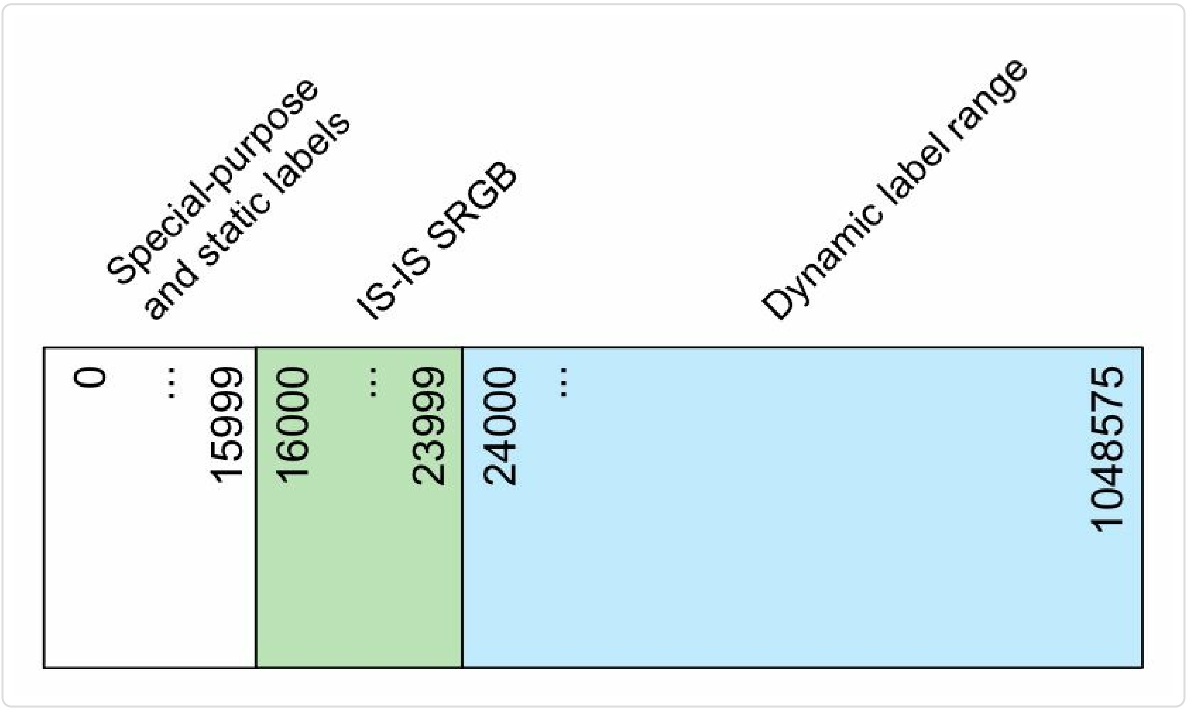


Figure 4-7: default SRGB allocation at boot time

Since IS-IS is the only priority LSD client that is activated on the system, all priority clients have now registered with LSD and their requested SRGBs are allocated. Now LSD can start allocating local labels for all MPLS Applications. For example, BGP requests a number of dynamic labels for L3VPN. LSD allocates these local labels starting at 24000.

Figure 4-8 illustrates the sequence of events to allocate a non-default SRGB at boot time. The sequence is equivalent to the one illustrated in Figure 4-7.

After the router image booted, LSD has carved the label range in the sections shows in the diagram:

- Labels from 0 to 15999 used for special-purpose and static labels
- Labels from 16000 to 23999 preserved for SRGB.
- Label range from 24000 to 1 million, used for dynamic labels.

LSD first waits for its Segment Routing protocol clients to register. Here, OSPF registers as a client with LSD and it has a non-default SRGB configured, from 30,000 to 39,999. Hence OSPF requests this local label range from LSD. LSD allocates this local label range for OSPF.

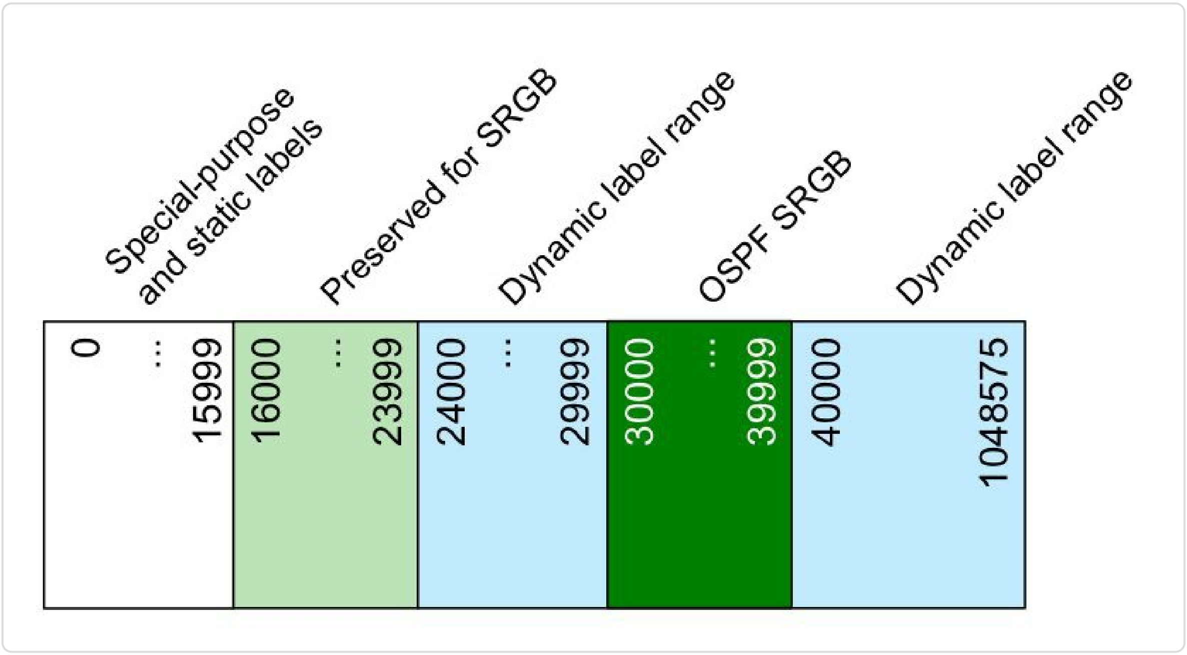


Figure 4-8: non-default SRGB allocation at boot time

Since OSPF is the only priority LSD client that is activated on the system, all priority clients have now registered with LSD and their requested SRGBs are allocated. Now LSD can start allocating local labels for all MPLS Applications. For example, BGP VPN requests a number of dynamic labels. LSD will allocate local labels starting at 24000 and outside of the SRGBs, since the labels in the SRGBs are already allocated for Segment Routing.

Figure 4-9 illustrates the mechanism to preserve the default SRGB local label range and how Segment Routing can be activated later on a running system. After the router image booted, LSD has carved the label range in the sections shows in the diagram. None of the LSD clients requested a SRGB local label range. In this initial state, no Segment Routing is enabled on the system, and no SRGB local label range is allocated. Still, LSD preserves the default SRGB label range 16,000 to 23,999. LSD allocated local labels for various MPLS applications starting at label 24,000.

Note that if an `mpls label range` configuration exists on the system and that label range includes (part of) the label range 16000 to 23999, then the default SRGB label range is *not* preserved. However, any label range that is not included in the configured dynamic label range will be preserved.

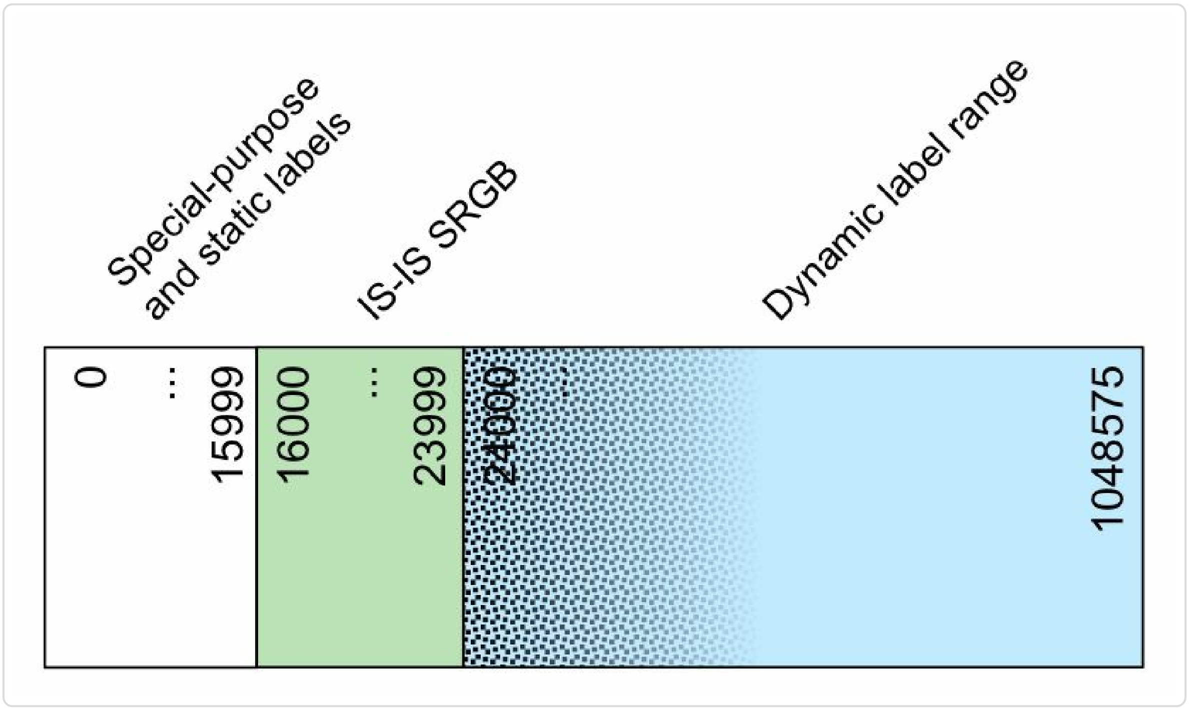


Figure 4-9: default SRGB allocation on running system

At this point Segment Routing is configured under `router isis`, using the default SRGB [16000-23999]. Since LSD has not allocated any label from that label range, that entire label range is unused. LSD can immediately allocate the SRGB label range for Segment Routing, the Segment Routing forwarding entries can immediately be installed and Segment Routing can immediately be used without requiring a reboot of the system.

"We will see in the further chapters of this book that Segment Routing can be enabled in an existing production network in a seamless and non-disruptive manner and services can be migrated on to it without service impact. SRGB allocation is perhaps the only exercise (and that too when not using the default) which may require a system reload. Hence, it is a good practice to plan and design the SRGB range and size in advance. If a non-default range is required, then it could be provisioned in a rolling manner (or whenever there is some planned or other maintenance activities on the routers requiring reboot) such that another reboot is specifically avoided whenever Segment Routing is actually enabled in the network."

— Ketan Talaulikar

4.5.2 SRGB Errors

If no Segment Routing forwarding entries (no prefix-SIDs, no Adjacency-SIDs) are being programmed in the forwarding table after enabling Segment Routing, it may be because the SRGB cannot be allocated. In that case, neither Prefix-SID labels nor Adjacency-SID labels are programmed in the forwarding table. An error message is logged in syslog to indicate SRGB allocation failed. The protocol periodically tries to allocate the SRGB. In Cisco IOS XR, the error is periodically logged until the SRGB allocation succeeds. See Example 4-10.

Example 4-10: IOS XR SRGB allocation failed message

```
RP/0/0/CPU0:Feb  5 09:30:21.181 : isis[1010]: %ROUTING-ISIS-4-
SRGB_ALLOC_FAIL : SRGB allocation failed: 'SRGB reservation not
successful for [24000,39999], srgb=(24000 39999,
SRGB_ALLOC_CONFIG_PENDING, 0x1) (So far 16 attempts). Make sure
label range is free'
RP/0/0/CPU0:xrvr-4#RP/0/0/CPU0:Feb  5 09:31:41.495 :
isis[1010]: %ROUTING-ISIS-4-SRGB_ALLOC_FAIL : SRGB allocation
failed: 'SRGB reservation not successful for [24000,39999],
srgb=(24000 39999, SRGB_ALLOC_CONFIG_PENDING, 0x1) (So far 32
attempts). Make sure label range is free'
...
```

The same error message is logged in Cisco IOS XR when an IGP instance tries to allocate a SRGB that is not the same but overlaps with the SRGB of another IGP instance. That is the case even if the SRGB is completely embedded within the other SRGB. The SRGBs must either be exactly the same or non-overlapping.

4.6 Summary

- Each node reserves a range of local labels, a Segment Routing Global Block (SRGB) that it uses for Global Segments.
- The default SRGB is a label range from 16000 to 23999, a non-default SRGB can be configured in any position between 16000 and 2^{20}-1.
- By default, Cisco IOS-XR preserves the 16000-23999 for SR and hence allows an eventual enabling of SR without requiring a reload.
- An SRGB can be made of multiple ranges but recommendation is to use a single range
- The recommended deployment model uses the same SRGB on each node.
- Formally, a Prefix SID is a unique index in the SRGB.
- The node originating the prefix advertises the index with the prefix SID.
- Any node in the SR domain derives the local label associated to the Prefix SID as SRGB base + index.
- In practice (homogenous SRGB) each node derives the same label from a Prefix-SID index (same SRGB base + same index = same label) and hence the label allocated to a Prefix SID is a global Label. For that reason, in practice, Prefix-SID in SR MPLS may also refer to the global allocated to the Prefix SID.

4.7 References

- [draft-filsfils-spring-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 Crabbe, E., "Segment Routing Use Cases", draft-filsfils-spring-segment-routing-use-cases-01 (work in progress), April 2015, https://tools.ietf.org/html/draft-filsfils-spring-segment-routing-use-cases-01.
- [draft-ietf-mpls-spring-lsp-ping] Kumar, N., Swallow, G., Pignataro, C., Akiya, N., Kini, S., Gredler, H., and M. Chen, "Label Switched Path (LSP) Ping/Trace for Segment Routing Networks Using MPLS Dataplane", draft-ietf-mpls-spring-lsp-ping (work in progress), May 2016, https://tools.ietf.org/html/draft-ietf-mpls-spring-lsp-ping.
- [draft-ietf-spring-segment-routing-mpls] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Shakir, R., Tantsura, J., and E. Crabbe, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls (work in progress), September 2016, https://tools.ietf.org/html/draft-ietf-spring-segment-routing-mpls.
- [draft-ietf-spring-segment-routing] Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and Shakir, R., "Segment Routing Architecture",draft-ietf-spring-segment-routing (work in progress), July 2016, https://tools.ietf.org/html/draft-ietf-spring-segment-routing.
- [draft-psarkar-spring-mpls-anycast-segments] Gredler, H., Filsfils, C., Previdi, S., Decraene, B., Horneffer, M., and Sarkar, P., "Anycast Segments in MPLS based Segment Routing", draft-psarkar-spring-

mpls-anycast-segments (work in progress), September 2016, https://tools.ietf.org/html/draft-psarkar-spring-mpls-anycast-segments.

5 SEGMENT ROUTING IGP CONTROL PLANE

The link-state Interior Gateway Protocols ISIS and OSPF have been extended to distribute Segment Routing information within the IGP domain along with the topology and reachability information that they currently signal. Each node in the IGP domain then uses this additional Segmenting Routing information in conjunction with its computed view of the network topology and prefix reachability information for setting up the forwarding tables. The forwarding state programmed by IGPs now also includes the MPLS labels corresponding to the Prefix and Adjacency Segments in a distributed manner across the network. This provides the foundation to steer traffic on any path though the network using Segments.

In this chapter, we will focus on the IGP control plane associated with the Segment Routing MPLS data plane and defer the description of the IPv6 data plane to a subsequent part of this book.

This chapter will also provide CLI configuration examples and sample CLI show command outputs to illustrate the working of these IGPs in Cisco IOS-XR software.

5.1 Segment Routing Extensions for IGPs

We begin with a high-level introduction of the extensions for the individual protocols for signaling the new Segment Routing information. Also, the very basic configurations to enable Segment Routing for each of the protocols.

In the further sections of this chapter, we will look more closely at aspects related to the signaling and processing of Segment Routing IGP control plane.

Extensions to the IGPs for handling SR/LDP interworking function, which allows for seamless deployment of Segment Routing in existing MPLS networks is covered separately in chapter 7, “Segment Routing and existing MPLS networks”.

5.1.1 Segment Routing for IS-IS

This section gives an overview of the ISIS Segment Routing control plane functionality. Segment Routing for ISIS supports both IPv4 and IPv6 address-families. It supports ISIS level-1, level-2 and multi-level routing.

In the Cisco IOS-XR implementation, Prefix-SIDs can currently be configured under ISIS for host prefixes (/32 for IPv4, /128 for IPv6) on loopback interfaces in the global routing table. ISIS will automatically allocate and advertise Adjacency-SIDs for all its adjacencies when that instance is enabled for Segment Routing.

5.1.1.1 IS-IS Protocol Extensions

ISIS is a highly extendable protocol; it does not use fixed format advertisements, but it uses Type/Length/Value (TLV) triplets to encode information in its advertisements. Sub-TLVs can be used inside a TLV to encapsulate further information elements. New protocol functionality can easily be added by defining new TLVs or extending existing TLVs.

The Segment Routing extensions for ISIS introduce the necessary sub-TLVs to attach Prefix-SIDs and Adjacency-SIDs to the various prefix and adjacency advertisement TLVs. The extensions also include the signaling of the support for Segment Routing capability and the SRGB in used by the router. The ISIS Segment Routing extensions are described in IETF draft-ietf-isis-segment-routing-extensions.

Segment Routing for IS-IS introduces support for the (sub-)TLVs in Table 5-1. The numbers between brackets are the (sub-)TLV types.

Table 5-1: ISIS Segment Routing extensions

Segment Routing (sub-)TLV	**Carried in TLV**
SR Capability sub-TLV (2)	IS-IS Router Capability TLV (242)
Prefix-SID sub-TLV (3)	Extended IP reachability TLV (135)
	SID/Label Binding TLV (149)
	Multi-Topology IPv4 IP Reachability TLV (235)
	IPv6 IP reachability TLV (236)
	Multi-Topology IPv6 IP reachability TLV (237)
Adjacency-SID sub-TLV (31)	Extended IS Reachability TLV (22)
	IS Neighbor Attribute TLV (23)
	Multi-Topology IS Reachability TLV (222)
	Multi-Topology IS Neighbor Attribute TLV (223)

LAN-Adjacency-SID sub-TLV (32)	Extended IS Reachability TLV (22)
	IS Neighbor Attribute TLV (23)
	Multi-Topology IS Reachability TLV (222)
	Multi-Topology IS Neighbor Attribute TLV (223)
SID/Label Binding TLV (149)	This is a top level TLV

The different ISIS Segment Routing TLVs and sub-TLVs are discussed in in more detail further in this chapter.

5.1.1.2 ISIS SR Configuration

The minimal configuration commands to enable Segment Routing for ISIS are displayed in Example 5-1.

Example 5-1: Enable MPLS Segment Routing for ISIS

```
router isis 1
 address-family ipv4|ipv6 unicast
  metric-style wide
  segment-routing mpls
```

Segment Routing functionality for ISIS requires wide metrics to be enabled for the address-family under the ISIS instance. This configuration enables advertisement of the TLV types that are also used by the Segment Routing control plane.

To enable Segment Routing for ISIS using the MPLS data plane, `segment-routing mpls` must be configured under the address family. MPLS Segment Routing is supported for both IPv4 and IPv6 address-families.

The `segment-routing mpls` configuration command enables both the Segment Routing control plane and the MPLS data plane. A number of actions are taken after enabling Segment Routing for MPLS under ISIS:

- ISIS allocates the SRGB and advertises the allocated SRGB in ISIS. Forwarding entries are programmed for any valid Prefix-SID received from other Segment Routing nodes in the network. The node advertises in ISIS for which address-families and data planes it can process Segment Routing packets. If the allocation of the SRGB fails then the SRGB and Segment Routing capabilities are still advertised, but no Segment Routing forwarding entries are programmed.
- ISIS automatically enables MPLS forwarding on all non-passive ISIS interfaces. MPLS labeled packets can then be transmitted and received over these interfaces. Note that MPLS forwarding is enabled on these interfaces even if no ISIS adjacency is actually established over the interface. If that is not desired then the interface should be configured as passive interface. Not enabling MPLS forwarding for passive interfaces is a safety precaution, to avoid that MPLS labeled packets can enter the network via such interface. Static MPLS can be used to enable MPLS on such interface if required.
- ISIS automatically allocates and advertises Adjacency-SIDs for all its adjacencies. IS-IS also programs the forwarding entries for its Adjacency-SIDs.

After configuring Segment Routing MPLS under ISIS, the router is ready to forward Segment Routing traffic. Advertisement of local prefix-SIDs requires additional configuration that will be covered later in this chapter.

Use the commands in Example 5-2 and Example 5-3 to verify if MPLS is enabled on the ISIS interfaces.

Example 5-2: SR MPLS enabled interfaces

```
RP/0/0/CPU0:xrvr-1#show mpls interfaces
Interface                  LDP      Tunnel   Static   Enabled
-------------------------- -------- -------- -------- --------
GigabitEthernet0/0/0/0     No       No       No       Yes
GigabitEthernet0/0/0/1     No       No       No       Yes
GigabitEthernet0/0/0/2     No       No       No       Yes
GigabitEthernet0/0/0/3     No       No       No       Yes
```

Example 5-3: SR MPLS enabled interface

```
RP/0/0/CPU0:xrvr-1#show mpls interfaces GigabitEthernet0/0/0/0
detail
Interface GigabitEthernet0/0/0/0:
        LDP labelling not enabled
        LSP labelling not enabled
        MPLS ISIS enabled
        MPLS enabled
```

HIGHLIGHT

Segment Routing can be enabled for an ISIS instance using a single command "`segment-routing mpls`" under the address family mode under "`router isis`" mode.

5.1.2 Segment Routing for OSPFv2

This section gives an overview of the Segment Routing control plane functionality in OSPFv2 which supports the IPv4 address-family. Segment Routing for OSPFv2 supports multi-area networks and provides the ability to enable Segment Routing for a specific area or all areas. In the Cisco IOS-XR implementation, Prefix-SIDs can be configured for host prefixes on loopback interfaces in the global routing table. OSPFv2 will automatically allocate Adjacency-SIDs for its adjacencies when Segment Routing is enabled for that area or at the instance level.

5.1.2.1 OSPFv2 Protocol Extensions

OSPFv2 was originally defined to use fixed length Link State Advertisement (LSA) for its base protocol operations. However, as the protocol evolved, Opaque LSAs were defined to allow for extensions to it for purposes like Traffic Engineering and other applications. These Opaque LSAs have since been extended to introduce new capabilities into OSPFv2 including Segment Routing.

REMINDER: OSPFv2 Opaque LSAs

OSPFv2 Opaque Link-State Advertisements (LSAs) are defined in IETF RFC 5250. The Opaque LSAs can be used directly by OSPF or by some other application that wishes to use OSPF to distribute information in the network. The Opaque LSAs may carry information that is not understood by transit OSPF routers (maybe running an older software version) – it would still flood it to its neighbors "opaquely". Only the OSPF routers that understand the information would actually use it. As such, Opaque LSAs provide a mechanism to extend the OSPF protocol in an existing network deployment without breaking existing functionality.

Opaque LSAs use three types of OSPFv2 LSAs (9, 10, and 11), each LSA type indicates a different flooding scope of the Opaque LSA. So, the functionality of these three LSA types is the same, they are all Opaque LSAs, only their flooding scope is different. The application that uses Opaque LSAs chooses which LSA type it uses, depending on how wide the data must be distributed in the network.

The flooding scope of Opaque LSAs:

- Type 9 LSA, a link-local scope Opaque LSA. This type of LSA is not flooded beyond the local (sub)network.
- Type 10 LSA, an area-local scope Opaque LSA. This type of LSA is not flooded beyond the area that they are originated into.
- Type 11 LSA, an AS-wide scope Opaque LSA. This type of LSA is flooded throughout the

Autonomous System (i.e. has the same scope as e.g. type-5 LSAs). Opaque LSAs with AS-wide scope are not flooded into stub areas.

Although TE uses Opaque LSAs (specifically type 10 LSAs) to distribute TE link attributes, Opaque LSAs should not be reduced to “TE LSAs”; TE is only one (but probably best-known) of the applications using Opaque LSAs. Opaque LSAs are used to distribute TE topology information (extended TE link attributes) in the network. TE only uses Type 10 LSAs, the Opaque LSAs with area flooding scope.

To allow different applications to use Opaque LSAs, they carry an “Opaque Type” field that indicates its type. The TE Opaque LSAs are using Opaque Type 1. Another example is the Router Information Opaque LSA: to advertise OSPF capabilities, an Opaque LSA of type 4 is used.

Format of an Opaque LSA as specified in IETF RFC 5250 is shown in Figure 5-1.

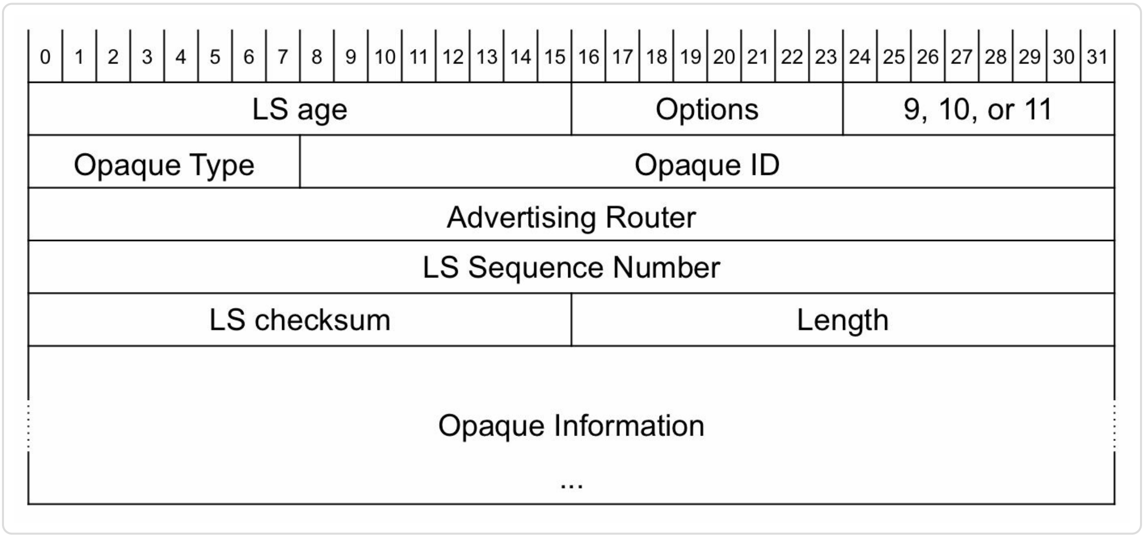


Figure 5-1: Opaque LSA format

The common 20-byte OSPF LSA Header contains a 32 bits Link-State ID field (bits 32-63), as defined in OSPFv2 IETF RFC 2328. For the Opaque LSAs this Link-State ID field is divided into an Opaque Type field (the first 8 bits) and a type-specific Opaque ID field (the remaining 24 bits). In the `show ospf database` output, these two fields are combined into the Link-State ID field and presented in a dotted decimal representation. It is displayed in the form of an IPv4 address but it has no IPv4 address semantics. The first digit in the Link-State ID field is the Opaque LSA Type, the last 3 digits is the Opaque ID.

Example 5-4 illustrates the output of show OSPF database for an Opaque LSA.

Example 5-4: Opaque LSA output example

```
 1| RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 4.0.0.0
self-originate
 2|
 3|            OSPF Router with ID (1.1.1.1) (Process ID 1)
 4|
 5|                Type-10 Opaque Link Area Link States
(Area 0)
 6|
 7|   LS age: 314
 8|   Options: (No TOS-capability, DC)
 9|   LS Type: Opaque Area Link
10|   Link State ID: 4.0.0.0
11|   Opaque Type: 4
12|   Opaque ID: 0
13|   Advertising Router: 1.1.1.1
14|   LS Seq Number: 80000075
15|   Checksum: 0x25d8
16|   Length: 52
17|
18|     Router Information TLV: Length: 4
19|     Capabilities:
20|       Graceful Restart Helper Capable
21|       Stub Router Capable
22|       All capability bits: 0x60000000
23| <...>
```

The type of LSA is indicated in line 5: `Type-10 Opaque Link Area`, which indicates an Opaque LSA with area flooding scope. The possibilities for Opaque LSAs are:

- Type-9 Opaque Link Local
- Type-10 Opaque Link Area
- Type-11 Opaque Link AS

The type of Opaque LSA is indicated in line 11: `Opaque Type: 4`, which indicates a Router Information Opaque LSA. Note that the Opaque Type has nothing to do with the flooding scope of the Opaque LSA, but it indicates which data is inside of the Opaque LSA.

The Opaque ID (line 12) is generally set to 0 for a Router Information Opaque LSA to indicate the first instance of this LSA (with RFC 7770 there can now be multiple).

The Opaque Type and Opaque ID are repeated in dotted decimal representation in the Link State ID (line 10): 4.0.0.0.

OSPFv2 has been extended to support advertisement of extended link and prefix attributes to allow signaling of additional information what is carried in the base OSPFv2 fixed length LSAs via RFC 7684. Following types and TLVs have been defined:

- OSPFv2 Extended Prefix Opaque LSA (Opaque-type 7)
  - OSPFv2 Extended Prefix TLV (type 1)
- OSPFv2 Extended Link Opaque LSA (Opaque-type 8)
  - OSPFv2 Extended Link TLV (type 1)

The generic OSPF extensions in IETF RFC 7684 were introduced based on the Segment Routing requirements, but they are also leveraged to carry other, non-SR related prefix and link attributes. This allows for a more general extension of the protocol.

Instead of introducing new Opaque LSA types, another option would have been to extend the TE Opaque LSAs to support Segment Routing. The reason why this option has not been chosen is that the TE Opaque LSAs have been defined to carry extended link attributes in the *Traffic Engineering* context. Following the specification of these TE LSAs in IETF RFC 3630, a link described in a TE Opaque LSA becomes part of the TE topology (which in most cases imply traditional RSVP-TE application), which is not always desired. Segment Routing should not be tied to traditional Traffic Engineering. The new Opaque LSA types that Segment Routing uses allow distributing link attributes free of the existing TE connotations.

The OSPFv2 Segment Routing extensions are specified in IETF draft-ietf-ospf-segment-routing-extensions. The Segment Routing information that is distributed in OSPF consists of attributes attached to prefixes and links. The Opaque-type 7 LSA is used to distribute Prefix-SIDs. Opaque-type 8 LSA is used to distribute Adjacency-SIDs. The Prefix-SIDs and Adjacency-SIDs are added respectively as sub-TLVs in the Extended Prefix TLV and the Extended Link TLV:

- OSPFv2 Extended Prefix Opaque LSA (Opaque-type 7)
  - OSPFv2 Extended Prefix TLV (type 1)
    - Prefix SID Sub-TLV (type 2)
- OSPFv2 Extended Link Opaque LSA (Opaque-type 8)
  - OSPFv2 Extended Link TLV (type 1)
    - Adj-SID Sub-TLV (type 2)
    - LAN Adj-SID Sub-TLV (type 3)

The Router Information Opaque LSA (Opaque-Type 4) is extended to carry the Segment Routing capabilities in OSPF. Two new TLVs are defined to include in the Router Information LSA:

- Router Information Opaque LSA (Opaque-Type 4)
  - SR-Algorithm TLV (8)
  - SID/Label Range TLV (9)

The new Opaque LSAs and the Segment Routing OSPFv2 extensions are described in detail in sections 5.3.4 “Prefix-SID Advertisement in OSPFv2” and 5.4.6.1 “OSPFv2 Adjacency-SID Advertisement”.

5.1.2.2 OSPFv2 SR Configuration

The simplest way to enable Segment Routing for OSPF on a node in Cisco IOS XR is to configure `segment-routing mpls` under the OSPF instance. This enables both the Segment Routing MPLS data plane and control plane functionality for all areas of this OSPF instance.

A number of actions are taken after enabling Segment Routing for MPLS under OSPF.

- OSPF allocates the SRGB and advertises the allocated SRGB in OSPF. Forwarding entries are programmed for any valid Prefix-SID received from other Segment Routing nodes in the network. If the allocation of the SRGB fails then the router will not advertise SRGB or Segment Routing capabilities so as not to attract Segment Routing traffic towards it. It will also not program Segment Routing forwarding entries.
- OSPF automatically enables MPLS forwarding on all non-loopback OSPF interfaces, even if no adjacency is established over the interface and even if the interface is configured as passive. MPLS labeled packets can then be transmitted and received over these interfaces. If required, Segment Routing MPLS forwarding can be disabled on specific interfaces with the configuration `segment-routing forwarding mpls disable` under the OSPF interface.
- OSPF automatically allocates and advertises Adjacency-SIDs for all its adjacencies on interfaces that are enabled for Segment Routing MPLS forwarding. OSPF also programs the forwarding entries for its Adjacency-SIDs.

After configuring Segment Routing MPLS under OSPF, the router is ready to forward Segment Routing traffic. Prefix-SIDs for local prefixes need to be configured to advertise them in OSPF. This is covered later in this chapter.

REMINDER: OSPF hierarchical configuration model

The OSPF configuration uses a hierarchical model. This is a brief reminder of the specifics of this OSPF configuration model. On top of the hierarchy is the OSPF instance or OSPF process; the level below is the OSPF area, and the lowest level is the OSPF interface. In this hierarchical model, the lower levels inherit configurations of a higher level. The level at which the command is finally applied is called the "scope" of the command.

For example, an OSPF configuration with interface scope, such as "hello-interval", which is configured at the OSPF instance level, is inherited by all interfaces of this OSPF instance. If an OSPF configuration is applied at multiple hierarchical levels, the lower level configuration (more specific) is preferred over the configuration done at a higher level (less specific). For example a configuration that enables an interface scope functionality on the OSPF instance level can be disabled on specific interfaces.

The Segment Routing configuration follows the usual OSPF configuration inheritance. The Segment Routing control plane and data plane functionality can be enabled on a single area by configuring the command `segment-routing mpls` under this area only. This command has area scope.

The Segment Routing data plane functionality can be disabled or enabled per interface with the command `segment-routing forwarding [disable | mpls]`. This command has interface scope. Note that `segment-routing forwarding mpls` is enabled by default when enabling Segment Routing.

If `segment-routing forwarding disable` is configured on an interface, then Adjacency-SIDs are not allocated, installed and distributed for that interfaces and the Prefix-SIDs of remote nodes, received by the Segment Routing control plane and that have a nexthop out of this interface, are not installed in the forwarding table.

In Example 5-5, Segment Routing is enabled for all interfaces in OSPF area 0, except for interface Gi0/0/0/0.

Example 5-5: OSPF: disable SR forwarding per interface

```
router ospf 1
 segment-routing mpls
 area 0
  interface GigabitEthernet0/0/0/0
   segment-routing forwarding disable
  !
 !
!
```

HIGHLIGHT

Segment Routing can be enabled for an OSPF instance using a single command `segment-routing mpls` under `router ospf` mode.

Use the hierarchical configuration model support to flexibly enable features at instance, area and interface levels.

5.1.3 Segment Routing for OSPFv3

On similar lines as OSPFv2, the OSPFv3 version of the protocol supports the IPv6 address family. As such, OSPFv3 can support Segment Routing for IPv6 using both MPLS and native IPv6 data plane. The Segment Routing Extensions are specified in IETF draft-ietf-ospf-ospfv3-segment-routing-extensions and uses the LSA extension infrastructure that is specified in IETF draft-ietf-ospf-ospfv3-lsa-extend.

The support for Segment Routing in OSPFv3 is not yet released in Cisco IOS-XR implementation and as such it would be covered in a future revision of this book.

"OSPF and ISIS are widely used routing protocols across IP networks and well understood by most network operators. With the help of a few SR extensions, the same protocols now support the enablement of MPLS and signaling of labels (along with benefits like automatic protection with TI-LFA). This has simplified MPLS for most operators that previously avoided MPLS due to the need for learning and provisioning of additional protocols like LDP and RSVP-TE. Leveraging of familiar IGP protocols is also one of the key factors for the rapid adoption of SR."

— *Ketan Talaulikar*

5.2 Segment Routing Capabilities

Each node advertises its Segment Routing capabilities in IGP. It advertises its support for Segment Routing, its Segment Routing data plane support, its support of Segment Routing algorithms and its SRGB.

5.2.1 SRGB Advertisement

The SRGB has local significance; it is a local block of labels allocated for Segment Routing Global Segments. However, all the nodes in the network need to know each other node's allocated SRGB, each other node's label range used for Segment Routing Global Segments. This is needed to calculate the label value that a node expects to receive for a Prefix Segment, the label value that is calculated from the SRGB base and the Prefix-SID index. Each node that supports Segment Routing Global Segments advertises its local SRGB in the routing protocol.

The configuration of the SRGB with different options and examples was already explained previously in chapter 4, "Management of the SRGB".

5.2.2 SR Capabilities Advertisement in IS-IS

An ISIS node advertises its Segment Routing data plane capability, its supported Segment Routing Algorithms, and its SRGB range using the IS-IS Router Capability TLV (TLV type 242, IETF RFC 4971).

The Router Capability TLV contains a Router-ID, Flags to indicate its flooding scope, and optional sub-TLVs. Segment Routing adds its capability sub-TLVs to this TLV.

The format of the Router Capability TLV is shown in Figure 5-2.

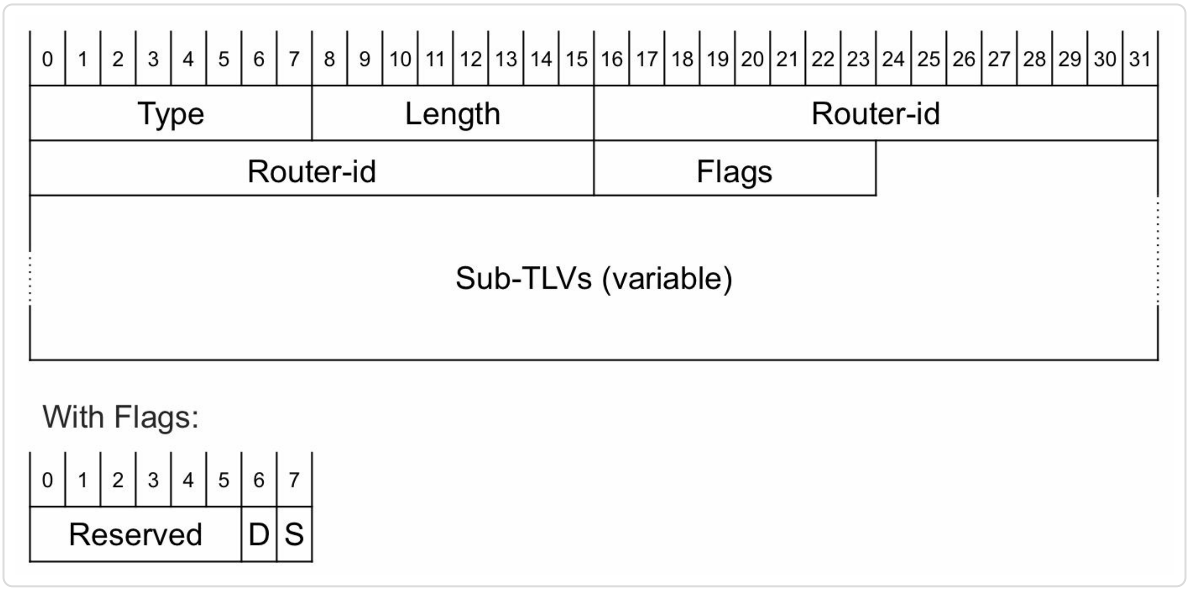


Figure 5-2: ISIS Router Capability TLV

ISIS uses a preference scheme to select the IP address that it advertises in the Router-id field of this TLV. ISIS looks subsequently for the following addresses on the node and the selection stops at the first prefix ISIS finds. In order of preference:

1. The configured MPLS TE router-id.
2. The configured ISIS router-id.
3. The IPv4 address of the lowest numbered Loopback interface configured under the ISIS instance
4. The IPv4 address of lowest numbered non-loopback interface configured under the ISIS instance

If no router-id is found, then the Router Capability TLV is not advertised.

HIGHLIGHT

It is strongly recommended to pick the host address (i.e. /32 prefix) of one of the loopback interfaces enabled in ISIS and to configure that specifically as the ISIS router-id. If MPLS TE is to be used then the same should be also configured as the MPLS TE router-id. This provides for a deterministic and stable router-id for operations.

Additionally, it is also strongly recommended to assign a Prefix-SID to this loopback interface (explained later in this chapter) that is used for router-id such that it can be used as the Node-SID for this router.

The Router Capability TLV flags:

- S (Scope): if set, then the TLV should be flooded across the entire routing domain
- D (Down): set if the TLV has been leaked from level-2 to level-1, unset otherwise. This flag is used to prevent unneeded flooding of the TLV. If this flag is set then the TLV is not propagated from level-1 to level-2.

The S-flag of the Router Capability TLV that carries the SR Capabilities sub-TLV is always unset, thus the SR Capabilities sub-TLV is not advertised across level boundaries. Consequently, the D-flag of this TLV is always unset as well.

When Segment Routing is configured under the IS-IS instance, the SR Capabilities sub-TLV (sub-TLV type 2) is added to the Router Capability TLV. This SR Capabilities sub-TLV indicates the Segment Routing data plane capability of the ISIS instance and the range of MPLS label values the ISIS instance uses for Global Segments (SRGB).

The format of the SR Capabilities sub-TLV is shown in Figure 5-3.

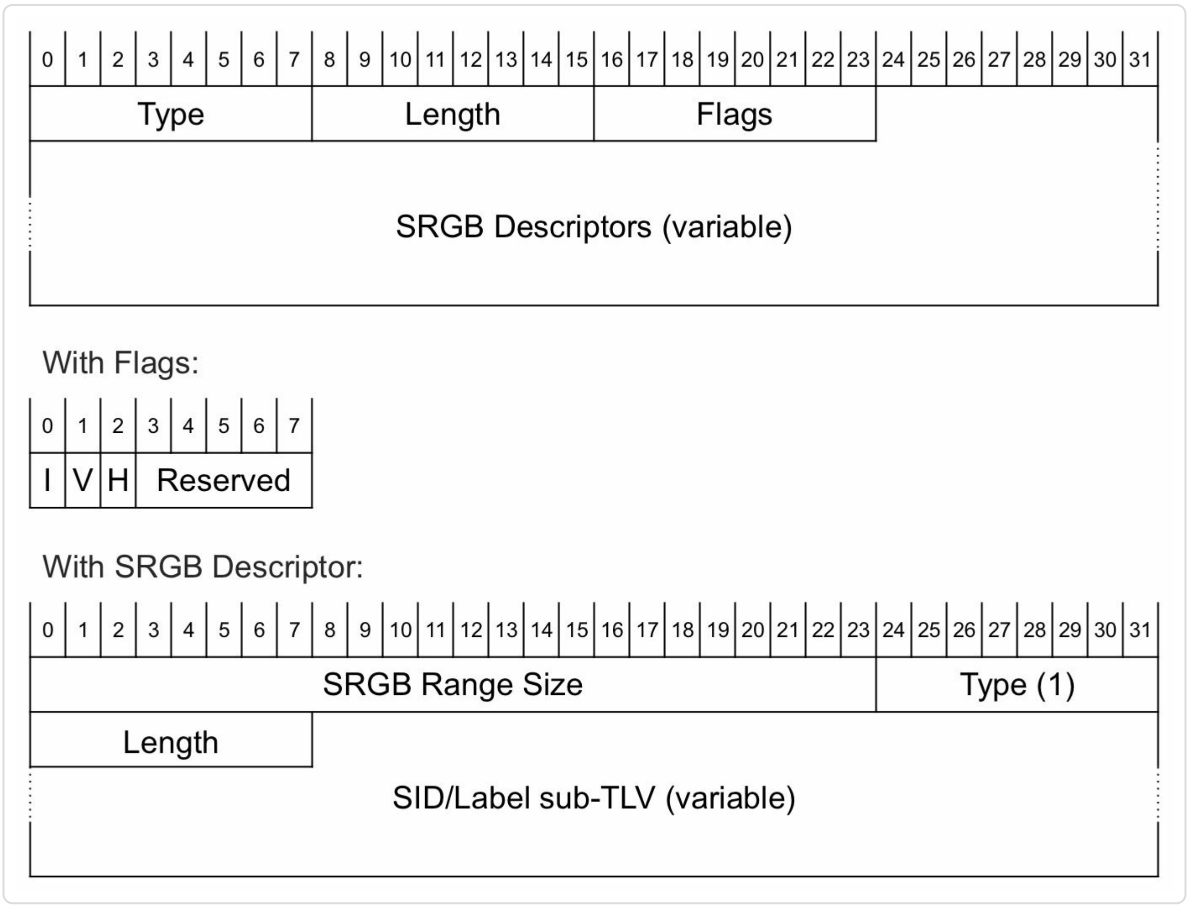


Figure 5-3: ISIS SR Capabilities sub-TLV format

The Flags field specifies:

- I (MPLS IPv4 flag): If set, then the node is capable of processing SR MPLS encapsulated IPv4 packets on all its interfaces
- V (MPLS IPv6): If set, then the node is capable of processing SR MPLS encapsulated IPv6 packets on all its interfaces
- H (SR-IPv6 flag): If set, then the node is capable of processing the IPv6 Segment Routing Header on all its interfaces

The SRGB Range Size in the SRGB Descriptor is a 24-bit field that contains the size of the SRGB label range, the number of labels in the SRGB. The SID/Label sub-TLV (type 1) specifies the SRGB base label value, the first label value in the SRGB label range. If the length of this sub-TLV is 3, then the rightmost 20 bits of the value represent an MPLS label.

Multiple SRGB ranges may be advertised, even if these ranges are disjoint, by adding more SRGB Descriptors in the SR Capabilities sub-TLV. These ranges are treated as an ordered set of ranges (in the order of advertisement) that are concatenated. SRGB with multiple ranges has been previously described in chapter 4, “Management of the SRGB”. At the time of writing this book, Cisco IOS XR does not support multiple SRGB ranges.

Another Segment Routing sub-TLV that may be advertised in the Router-capability TLV is the SR-Algorithm Sub-TLV (type 19). A node may use various algorithms to calculate reachability to other nodes and to prefixes attached to these nodes. Examples of these algorithms are the regular metric based Shortest Path First (SPF), Strict SPF, etc. The node advertises the algorithm(s) that it is currently using, in the SR-Algorithm sub-TLV. The format of the SR-Algorithm sub-TLV is shown in Figure 5-4. The Algorithm fields in this sub-TLV contain the numerical value identifying an algorithm. Each advertised Prefix-SID contains an Algorithm field with a value that refers to one of the algorithms listed in the SR-Algorithm sub-TLV. The Prefix-SID will be covered later in this section. The SR-Algorithm sub-TLV is optional. If it’s not advertised, then support of only algorithm 0 (metric based SPF) is assumed.

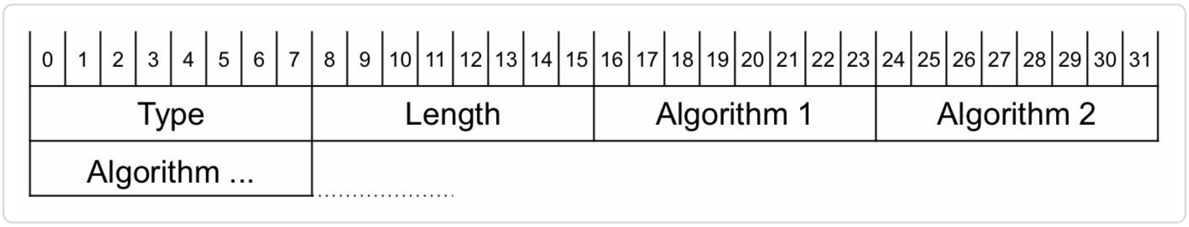


Figure 5-4: ISIS SR Algorithm sub-TLV format

At the time of writing this book, a single algorithm type was supported in ISIS: the well-known metric based Shortest Path First (SPF) algorithm (type 0) and no SR-Algorithm sub-TLV was advertised in the Router-capability TLV. However, as other algorithms (e.g. Strict SPF) support is added, the SR Algorithm TLV would get advertised mandatorily indicated support for both the SPF algorithm type 0 and the additional supported algorithms.

The network topology in Figure 5-5 is used to illustrate the advertisement of the Segment Routing capabilities in ISIS. The relevant ISIS configuration of Node1 is shown in Example 5-6. Segment Routing is enabled under IS-IS for both IPv4 and IPv6 address-families of both nodes. By default IPv4/IPv6 multi-topology mode (IETF RFC 5120) is enabled in IOS XR, where IPv4 and IPv6 are advertised as different (possibly non-congruent) topologies.

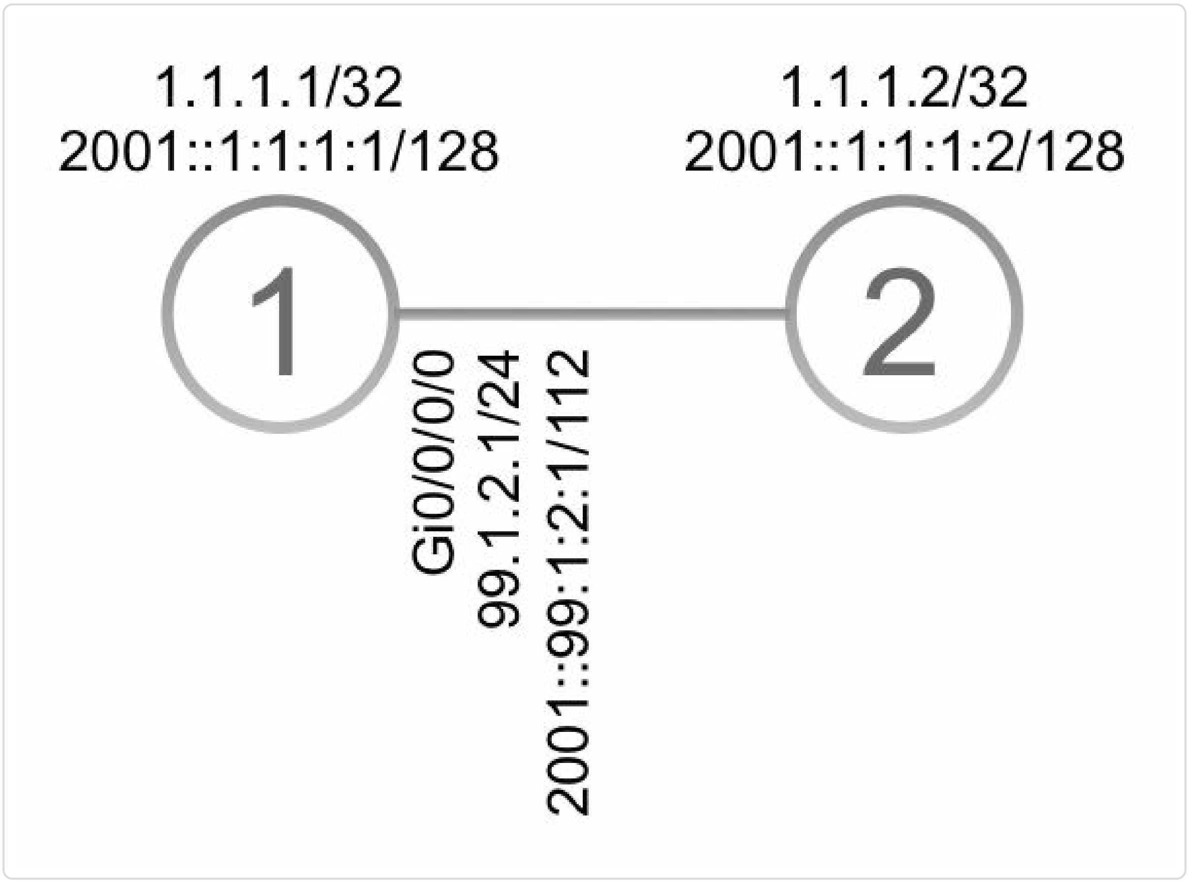


Figure 5-5: Network topology to illustrate ISIS SR Capabilities advertisement

Example 5-6: Multi-topology Segment Routing configuration example, Node1

```
interface Loopback0
 ipv4 address 1.1.1.1 255.255.255.255
 ipv6 address 2001::1:1:1:1/128
!
interface GigabitEthernet0/0/0/0
 description to xrvr-2
 ipv4 address 99.1.2.1 255.255.255.0
 ipv6 address 2001::99:1:2:1/112
!
router isis 1
 is-type level-2-only
 net 49.0001.0000.0000.0001.00
 address-family ipv4 unicast
  metric-style wide
  segment-routing mpls
 !
 address-family ipv6 unicast
  metric-style wide
```

```
   segment-routing mpls
  !
  interface Loopback0
   passive
   address-family ipv4 unicast
   !
   address-family ipv6 unicast
   !
  interface GigabitEthernet0/0/0/0
   point-to-point
   address-family ipv4 unicast
   !
   address-family ipv6 unicast
   !
  !
 !
```

The output of `show isis database verbose` for the LSP advertised by Node1 is shown in Example 5-7. The output shows that both IPv4 and IPv6 address families are enabled: Network Layer Protocol Identifiers (NLPIDs) 0xcc and 0x8e are advertised, for IPv4 and IPv6 respectively. IPv4 and IPv6 are enabled in Multi-Topology (MT) mode, which is the default in IOS XR: "MT: Standard (IPv4 Unicast)" and "MT: IPv6 Unicast" in the output. The Router Capability TLV is highlighted in Example 5-7. No router-id is explicitly configured on Node1, therefore the address of Node1's loopback0 interface, 1.1.1.1, is included in the Router-id field of the Router Capability TLV. The S-flag (Scope) is unset, which means that this Router Capability TLV must not be propagated between levels. Consequently the D-flag (Down), that indicates if this TLV has been propagated from Level-2 to Level-1, is also unset.

The Segment Routing Capabilities sub-TLV indicates that Node1 supports Segment Routing on both IPv4 over MPLS (I-flag=1) and IPv6 over MPLS (V-flag=1) data planes on all interfaces.

There is a single SRGB label range advertised, with Base label value 16000 and Range size 8000, which is the default SRGB [16000-23999].

Note that the SRGB is not address-family specific and indicates MPLS data plane for Segment Routing; the same SRGB is used for both IPv4 and IPv6 topologies.

Example 5-7: ISIS Router Capabilities advertisement example – Multi-topology

```
RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-1
IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x0000039b   0xfc27
1079            0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6
Unicast                                     0/0/0
  Hostname:     xrvr-1
  IP Address:   1.1.1.1
  IPv6 Address: 2001::1:1:1:1
  Router Cap:   1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
<...>
```

Example 5-8 shows the ISIS configuration of Node1 for the case where ISIS is running IPv4 and IPv6 address families in Single-topology mode. The only difference with the configuration in Example 5-6 is the `single-topology` command under address-family ipv6 unicast. In Single-topology mode the IPv4 and IPv6 topologies are congruent. The first few TLVs in the LSP advertised by Node1 are as displayed in Example 5-9. Single-topology or Multi-topology makes no difference for the Segment Routing capability advertisement. It will make a difference for other Segment Routing sub-TLVs, as will be indicated later in this section.

Example 5-8: Single-topology Segment Routing configuration example

```
router isis 1
 is-type level-2-only
 net 49.0001.0000.0000.0001.00
 address-family ipv4 unicast
  metric-style wide
  segment-routing mpls
 !
 address-family ipv6 unicast
  metric-style wide
  single-topology
  segment-routing mpls
 !
 interface Loopback0
  address-family ipv4 unicast
!
```

Example 5-9: ISIS Router Capabilities advertisement example – Single-topology

```
IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x00000976   0xbaca
1192             0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  Hostname:     xrvr-1
  IP Address:   1.1.1.1
  IPv6 Address: 2001::1:1:1:1
  Router Cap:   1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
<...>
```

5.2.3 SR Capabilities Advertisement in OSPFv2

OSPFv2 advertises the node’s Segment Routing capabilities in the Router Information Opaque LSA (defined in IETF RFC 7770). This OSPFv2 Router Information LSA has an Opaque-type 4 and would generally use Opaque ID 0. These LSAs are displayed in the OSPF database output with Link-State ID 4.0.0.0. Opaque LSAs can have different flooding scopes (link, area, or autonomous system). Distribution of the Segment Routing capabilities requires area scope flooding for the Router Information LSA, hence Type 10 LSAs (area flooding scope) are used. Since OSPFv2 only supports the IPv4 address-family, there is no need for flags to indicate the Segment Routing data plane support for OSPFv2. The SRGB is advertised in one or more SID/Label Range TLVs included in the Router Information Opaque LSA.

The format of the SID/Label Range TLV is shown in Figure 5-6.

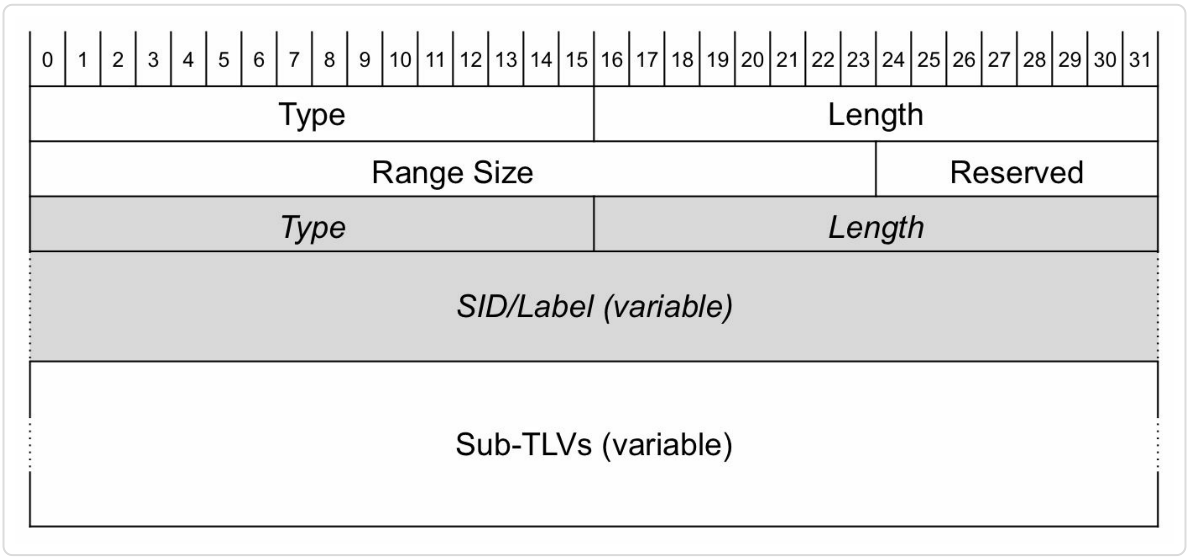


Figure 5-6: OSPFv2 SID/Label Range TLV format

The 24-bit Range Size field contains the size of the SRGB label range, the number of labels in the SRGB.

The SID/Label Range TLVs can contain a variable number of sub-TLVs. At the time this book was written, only the SID/Label sub-TLV was accepted as sub-TLV of the SID/Label Range TLV. Therefore this sub-TLV is displayed in the SID/Label Range TLV format in Figure 5-6. The SID/Label field in this sub-TLV represents the SRGB Base label value, the first label value of the advertised label range.

A SRGB can be composed of multiple, possibly disjoint, label ranges. To advertise such SRGB ranges, multiple SID/Label Range TLVs can be added to the Router Information LSA. These ranges are treated as an ordered set (in the order of advertisement), and the SID index is an offset in the set of ranges concatenated in the received order. At the time of writing this book, IOS XR does not support multiple SRGB ranges. See chapter 4, “Management of the SRGB” for an example of an SRGB consisting of multiple ranges.

A node may use various algorithms to calculate reachability to other nodes or to prefixes attached to these nodes. Examples of these algorithms are metric based Shortest Path First (SPF), Strict SPF, etc. A node advertises the algorithm(s) that it is currently using in the SR Algorithm TLV.

The format of the SR Algorithm TLV is shown in Figure 5-7.

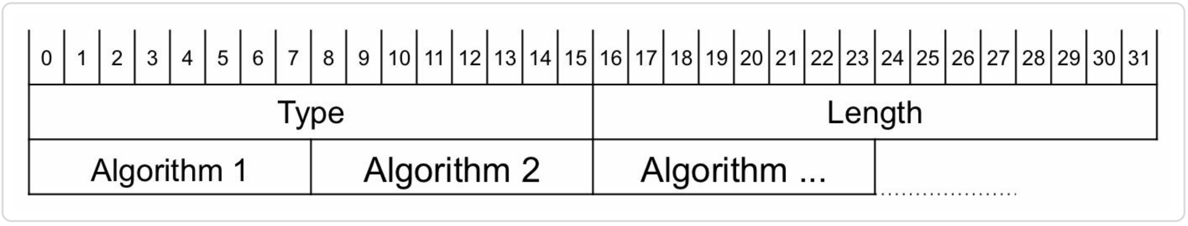


Figure 5-7: OSPFv2 SR Algorithm TLV format

The Algorithm fields contain the numerical value identifying an algorithm. As will be covered later in this section, each advertised Prefix-SID contains an Algorithm field with a value that refers to one of the algorithms listed in the SR-Algorithm sub-TLV. At the time of writing this book, OSPF supported the metric based SPF algorithm (type 0) and the strict SPF algorithm (type 1). The strict SPF algorithm is described briefly in chapter 2 but it will be covered in further detail in the next part of this book, along with its application use-cases.

The network topology in Figure 5-8 is used to illustrate the advertisement of the Segment Routing capabilities in OSPFv2. The relevant OSPF configuration of Node1 is shown in Example 5-10. Segment Routing is enabled under OSPFv2 for both nodes in the network.

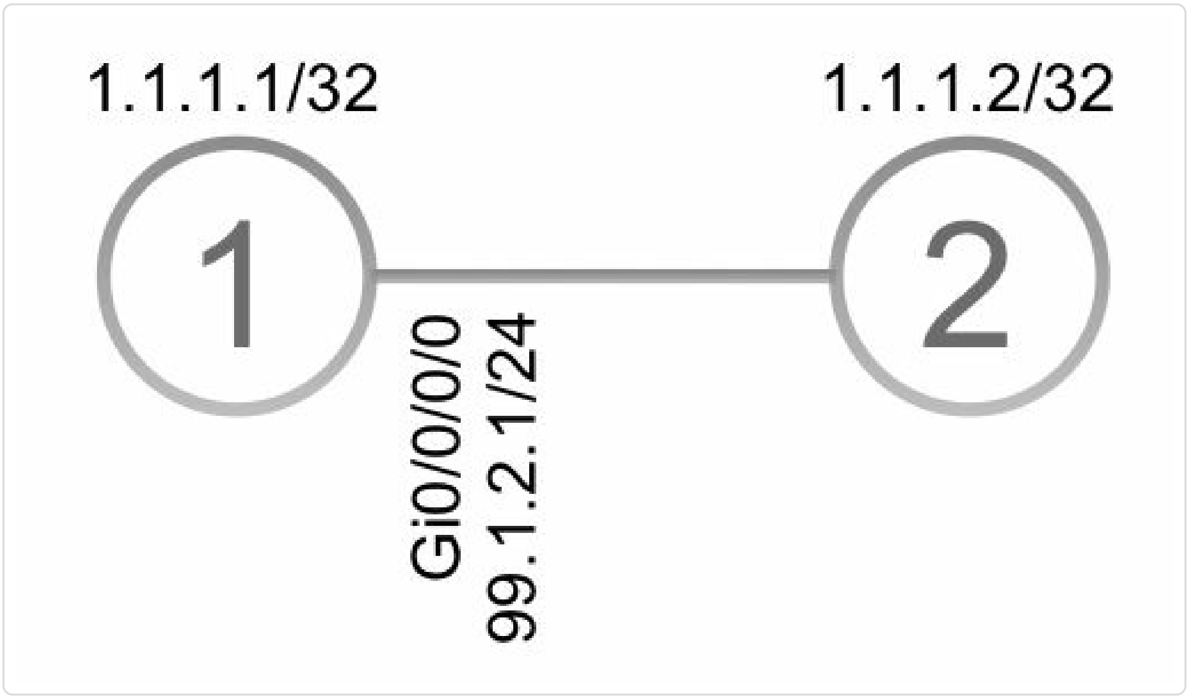


Figure 5-8: Network topology to illustrate OSPFv2 SR Capabilities advertisement

Example 5-10: OSPFv2 Segment Routing configuration example

```
router ospf 1
 router-id 1.1.1.1
```

```
  segment-routing mpls
  !! by default: segment-routing forwarding mpls
  area 0
   interface Loopback0
  !
 !
```

The output of `show ospf database opaque-area` of the Router Information Opaque LSA with Link-State ID 4.0.0.0 (Opaque Type 4 and Opaque ID 0.0.0) advertised by Node1 is displayed in Example 5-11. This LSA contains, besides the Router Information TLV, the Segment Routing Algorithm TLV and the Segment Routing Range TLV. The LSA is of type 10, an Opaque LSA with area flooding scope.

The Segment Routing Algorithm TLV contains the well-known metric based SPF algorithm (Algorithm 0) and the strict SPF algorithm (Algorithm 1).

The Segment Routing Range TLV contains a single SRGB of Range Size 8000 and SRGB Base value 16000. This is the default SRGB [16000-23999].

Example 5-11: OSPFv2 Router Information LSA example

```
RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 4.0.0.0 self-
originate

            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 0)

  LS age: 314
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 4.0.0.0
  Opaque Type: 4
  Opaque ID: 0
```

```
Advertising Router: 1.1.1.1
LS Seq Number: 80000075
Checksum: 0x25d8
Length: 52

  Router Information TLV: Length: 4
  Capabilities:
    Graceful Restart Helper Capable
    Stub Router Capable
    All capability bits: 0x60000000

  Segment Routing Algorithm TLV: Length: 2
    Algorithm: 0
    Algorithm: 1

  Segment Routing Range TLV: Length: 12
    Range Size: 8000

      SID sub-TLV: Length 3
       Label: 16000
```

HIGHLIGHT

It is strongly recommended to pick the host address (i.e. /32 prefix) of one of the loopback interfaces enabled in OSPF and to configure that specifically as the OSPF router-id. If MPLS TE is to be used then the same should be also configured as the MPLS TE router-id. This provides for a deterministic and stable router-id for operations.

Additionally, it is also strongly recommended to assign a Prefix-SID to this loopback interface (explained later in this chapter in section 5.3.2, “Prefix-SID Configuration”) that is used for router-id such that it can be used as the Node-SID for this router.]

5.3 Prefix-SID

The description of Prefix-SID and its different variants has been previously covered in chapter 2, “Fundamentals of Segment Routing”. Also, the key concepts of SRGB, Prefix-SID index, determination of label and the recommended scheme of using homogenous SRGB and global labels for Prefix-SIDs have already been covered in detail in chapter 3, “Segment Routing MPLS Data Plane” and chapter 4, “Management of the SRGB”. In this section, we will focus on how Prefix-SIDs are configured under IGPs and the IGP mechanisms related to their advertisement and programming of corresponding forwarding entries.

HIGHLIGHT: SRGB and Global Prefix-SID for SR MPLS

Formally, a Prefix-SID is a unique index in the SRGB.

The node originating the prefix advertises the index with the prefix-SID

Any node in the SR domain derives the local label associated to the Prefix-SID as SRGB_base + index

The recommended deployment model calls for homogenous SRGB through the SR domain

In practice (using homogenous SRGB), each node derives the same label (same SRGB base + same index = same label) and hence the label allocated to a Prefix-SID is a Global Label. For that reason, in practice, Prefix-SID in SR MPLS may also refer to the Global Label allocated to the Prefix.

5.3.1 Signaling Prefix-SID in IGP Advertisements

The base IGP protocols today already signal Prefix reachability information along with their topological information. Every IGP node then computes the topology and prefix reachability from its perspective and programs the forwarding entries corresponding to the reachable prefixes. It would seem to be a natural extension for the IGPs to also signal the MPLS labels for these prefixes rather than requiring the use of an additional protocol like LDP. The resulting simplicity and efficiency would seem obvious.

During the very initial days of Segment Routing development, it was proposed that IGPs actually signal the specific MPLS global label associated with the Prefix. This later evolved to what we have today – i.e. IGPs nodes advertise their SRGBs (giving the label range) and then for each prefix advertise the SID in the form of an index. Some of the history and reasoning for this change has been described in the opening chapter of this book.

In the previous chapters “Segment Routing MPLS Data Plane” and “Management of the SRGB”, we saw how the MPLS label corresponding to the Prefix-SID is determined from the SRGB and the Prefix-SID index and how we can get a truly Global Label for a Prefix-SID when using a homogenous SRGB in the network.

Another aspect of doing Prefix-SID and its label signaling via IGPs is that only the necessary prefixes (e.g. the router-id loopbacks, service loopbacks, etc.) need to get labels assigned for them and all other prefixes which do not have any service traffic destined to them are skipped without any complex filtering. This also naturally reduces the scale for the MPLS forwarding entries in the network.

With Segment Routing extensions, we will see further in this chapter how IGPs additionally distribute the Prefix-SID information along with their existing prefix distribution mechanism.

HIGHLIGHT

The classic MPLS control-plane uses LDP to distribute the labels associated with the IGP prefixes. This is a redundant and complex design decision.

In SR, straightforward IGP extensions are implemented to distribute these labels natively in the IGP.

Eliminating LDP means less code to maintain and qualify, less software bugs to experience and troubleshoot, and less performance drain on the distributing routing system.

Eliminating LDP means simpler SDN operation as the controller only needs to retrieve the IGP link-state topology to infer the topology and the related segments.

Eliminating LDP means eliminating the infamous LDP/synch problem (IETF RFC 5443)

“The objective of the SR project was absolutely *not* to eliminate LDP or RSVP-TE.

The objective was to simplify the operations of IP networks, increase their scale and functionality and finally enable applications to control the network without imposing flow state all along the infrastructure.

The out-of-the-box SR research resulted in a completely different control-plane for the MPLS data-plane.

A side-effect of that analysis was to realize that LDP and RSVP-TE were not needed.

Again, this was a side-effect of the analysis, not an objective.”

— Clarence Filsfils

5.3.2 Prefix-SID Configuration

Since Prefix Segments are Global Segments, they must be globally uniquely identifiable. “Global” means “domain wide” here. Each Global Segment must have a globally unique SID. In the MPLS implementation of Segment Routing, Global Segments are identified by a globally unique SID index. This global SID index points to a label in the local SRGB.

An IGP Prefix-SID is not automatically allocated by a node. It must be explicitly configured such that its global uniqueness can be ensured. The Segment Routing specification allows attaching Prefix-SIDs to non-host prefixes, but the current Cisco IOS XR implementation only allows configuration of Prefix-SIDs attached to a /32 or /128 global prefix configured on a loopback interface. We have discussed previously in chapter 2, “Fundamentals of Segment Routing” on how these Prefix-SIDs can be provisioned for select loopback prefixes in the network, their requirement for domain-wide uniqueness and how they can be provisioned in a manner that avoid conflicts and errors.

In the event of errors in provisioning or conflicts, syslog and other error messages are generated and the concerned Prefix-SIDs will not become operational. There are ongoing standardization discussions within the IETF SPRING working-group at the time of writing this book[1], which attempt to make conflict determination and resolution a deterministic process with minimal impact on existing network services. We will likely update the conclusions on those efforts in a further revision of this book.

In the Prefix-SID index configuration, the configured index is the globally unique index that is allocated for that Prefix Segment. The index is zero-based, i.e. 0 is the first index, and represents the offset in the SRGB starting from the SRGB Base, the first label value in the SRGB. To calculate the absolute local label value for the Prefix-SID from the SID index, add the SID index to the SRGB Base value.[2] For example, SID index 1 with the default SRGB, which starts at label 16000, gives as local label value for this SID 16000+1=16001.

If the recommended homogenous SRGB deployment model is followed, then the local label value for this Prefix-SID index is the same on all nodes and Prefix-SIDs have global label *values*. This provides significant operational benefits. An ability to configure the Prefix-SID as an absolute label value extends this Prefix Segment global label *look-and-feel* to the router configuration. When the Prefix-SID is configured as an absolute label value then that absolute label value matches the label found in the forwarding tables for that Prefix Segment, even though the underlying IGP protocol signaling is still index based (as we will see further in this section). This extends the simplicity of the deterministic and intuitive label allocation of SR to the router configuration, along with the homogenous SRGB model. Leading operators that were involved in the early development of Segment Routing preferred this simple model. The configuration model for Prefix-SID on Cisco IOS-XR software hence provides a choice for configuring either as an index value or as an absolute (global) MPLS label value.

Each node in the Segment Routing IGP Domain should have a Prefix-SID configured under its loopback interface that is enabled in the IGP. The prefix of that loopback interface is only originated by that node; typically this would be the same host prefix (IPv4 /32 or IPv6 /128) that is configured as the “router-id”. Such a Prefix Segment would be also called as a “Node Segment” since it uniquely identifies a node in the network.

The N-flag (Node flag) is always set for a Prefix-SID by default to indicate it is a Node-SID.

This might not be viewed as a strict requirement if the node only acts as a Segment Routing mid-point node; if the node only carries Segment Routing transit traffic. But to steer traffic *to* a node or *via* a node using Segment Routing Prefix Segments, then this node must have a Prefix-SID configured and must advertise it. Also, as we will see further in chapter 9, “Topology Independent LFA (TI-LFA)”, this is required to be done on all nodes in order to ensure that TI-LFA feature that provides automatic backup paths for all prefixes operates correctly.

Example 5-12 and Example 5-13 show the Prefix-SID configuration for ISIS and OSPF in Cisco IOS XR. Prefix-SIDs can be configured as absolute label values within the SRGB (“absolute Prefix-SID” mode) or as an index (offset) within the SRGB. Inconsistent values are either rejected with errors at the time of configuration or in some cases the errors are flagged via syslog messages and the Prefix-SID is not used until the error is corrected.

The recommended model to deploy and operate a Segment Routing network is the “global label *value*” mode or “absolute Prefix-SID” mode. In this “absolute Prefix-SID” mode, all nodes in the network use the same SRGB.

Example 5-12: ISIS Prefix-SID configuration

```
router isis 1
 interface Loopback0
  address-family ipv4|ipv6 unicast
   prefix-sid (absolute|index) (<SID value>|<SID index>)
[explicit-null] [n-flag-clear]
```

Example 5-13: OSPF Prefix-SID configuration

```
router ospf 1
 area 0
  interface Loopback0
   prefix-sid (absolute|index) (<SID value>|<SID index>)
[explicit-null] [n-flag-clear]
```

If not all nodes use the same SRGB, then Prefix-SIDs are better configured as indexes because in that case a Prefix-SID will not have the same label value on all nodes. Since labels are not global, configuring different absolute labels across routers can get confusing for the user and increase the chances of causing conflicts which can be avoided by using unique index values that avoid conflicts across nodes.

OSPF also provides support for configuring Prefix-SID for the Strict SPF algorithm usage with the `strict-spf` optional keyword. However, we will skip further description for this algorithm along with its use-cases for the next part of this book.

5.3.2.1 Prefix-SID with N-flag Unset

A Node-SID can be distinguished from a regular Prefix-SID in the IGP advertisement, by the setting of the N-flag: a Node-SID is a Prefix-SID with the N-flag set. By default, the N-flag is set for any configured Prefix-SID, i.e. any configured Prefix-SID is a Node-SID by default. However, there are cases (e.g. Anycast-Segment) where the same Prefix-SID is being advertised by multiple nodes and hence is not really a Node-ID; such Prefix-SID must be configured with the N-flag unset. Use the `n-flag-clear` keyword in the Prefix-SID configuration command to clear the N-flag. See the ISIS and OSPF Prefix-SID configuration syntax in Example 5-12 and Example 5-13.

We have seen previously in chapter 2, “Fundamentals of Segment Routing” the concept and usage of Anycast-SID and also in chapter 4, “Management of the SRGB” the homogenous SRGB requirement for using it. There is actually nothing in the Prefix-SID advertisement that indicates it is an Anycast-SID. However, it is important that Anycast-SIDs are not configured *without* the `n-flag-clear` option as they could get misconstrued to be a Node-SID and as such cause problems with features like TI-LFA, which rely on Node-SIDs being unique in the domain. It is, however, not a problem in configuring the `n-flag-clear` option on Prefix-SIDs that are not Anycast-SIDs, as long as there is at least one Node-SID provisioned on the router for ensuring that features like TI-LFA work properly.

5.3.2.2 Prefix-SID Penultimate Hop Behavior

The node originating a Prefix Segment specifies the behavior of the penultimate (“next-to-last“) hop node for that Prefix Segment. The penultimate hop node is the neighbor node that is upstream from the originator of the Prefix-SID. Two flags are available in the IGP Prefix-SID advertisement to indicate the required penultimate hop behavior: the PHP-off flag and the Explicit-null flag. The PHP-off flag indicates if the penultimate hop should pop the label or not before forwarding to the final node. Note the “-off” in “PHP-off”, if the PHP-off flag is not set, then it means that PHP is requested. This flag has the name “P-flag” in ISIS and “NP-flag” in OSPF.

No configuration is required to request the Penultimate Hop Popping (PHP) behavior. By default a prefix-SID is advertised with the PHP-off flag unset.

If the originator of a Prefix-SID requires receiving the packets on the Prefix Segment with the MPLS Traffic Class field available, then it can request the explicit-null behavior from the penultimate hop. In that case the Prefix-SID originator advertises the Prefix-SID with both the Explicit-Null flag (E-flag) and PHP-off flag set. Note that the PHP-off flag must be set if the Explicit-Null flag is set.

Add the `explicit-null` keyword to the configuration of the Prefix-SID to request the explicit-null behavior for this Prefix-SID. See the ISIS and OSPF Prefix-SID configuration syntax in Example 5-12 and Example 5-13. By default the Explicit-Null flag and PHP-off flags are not set.

The Segment Routing extensions also allow to request PHP-off behavior for the penultimate hop node. In that case the penultimate hop node will forward the packets on the Prefix Segment to the final node with the Prefix-SID label on top. The final node then has to do a double lookup: on the top Prefix-SID label to find out it is its local Prefix-SID and then on the label or header underneath. This behavior is not configurable in Cisco IOS XR.

HIGHLIGHT

It is recommended to configure the “router-id” as a valid reachable loopback host prefix (i.e. /32 prefix) which is also advertised by the IGP; the same should also have Prefix-SID configured that would become the Node-SID.

Additional Prefix-SIDs can be configured for other service loopback addresses or for Anycast-SID usage; Anycast-SIDs must be always configured with n-flag-clear option

IGPs automatically determine and use the optimal PHP behavior; explicit-null option is to be used only in designs where the TC/EXP bits in MPLS label needs to be conveyed to the destination via the MPLS label.

5.3.3 Prefix-SID Advertisement in IS-IS

To advertise a Prefix-SID associated with a prefix, ISIS attaches a Prefix-SID sub-TLV to the prefix reachability advertisement. As shown in Table 5-1, it can be included in one of the following IP Reachability TLVs: TLV-135 (IPv4), TLV-235 (MT-IPv4), TLV-236 (IPv6), and TLV-237 (MT-IPv6).

The Prefix-SID sub-TLV has the format as shown in Figure 5-9.

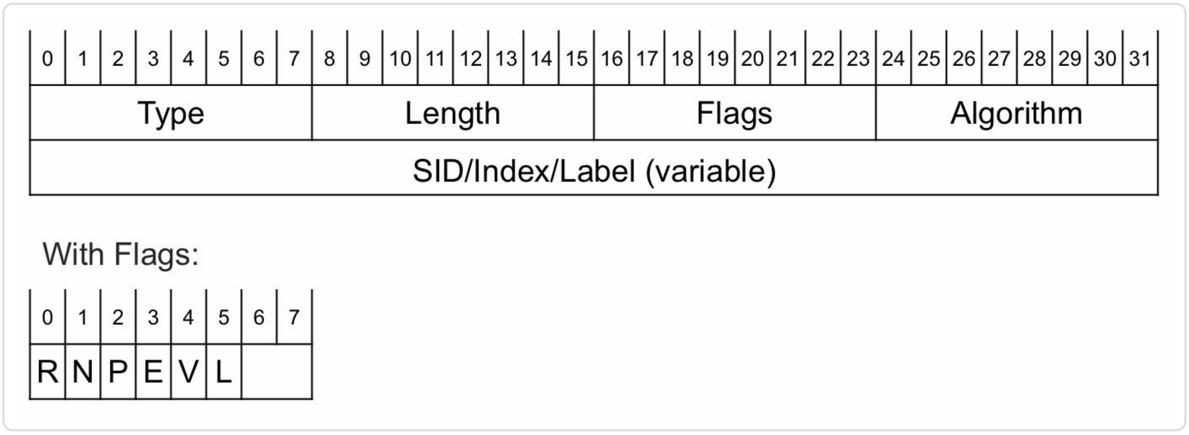


Figure 5-9: ISIS Prefix-SID sub-TLV format

- The flags in this sub-TLV are:
  - R (Re-advertisement): set if the prefix attached to this Prefix-SID is a non-local prefix that has been propagated from another level or redistributed from another protocol – default in Cisco IOS XR: unset
  - N (Node-SID): set if the Prefix-SID is a Node-SID, i.e. identifies the node – default in Cisco IOS XR: set
  - P (PHP-off): set if the penultimate hop must NOT pop the prefix-SID before forwarding the packet – default in Cisco IOS XR: unset, i.e. PHP is on by default
  - E (Explicit-Null): set if penultimate hop must swap prefix-SID with

    Explicit-Null label – default in Cisco IOS XR: unset

  - V (Value): set if prefix-SID carries a label value (not an index) – Cisco IOS XR: always unset
  - L (Local): set if prefix-SID has local significance – Cisco IOS XR: always unset

The N-flag and R-flag are generic prefix properties or prefix attributes, not specific to Segment Routing. These flags are also carried by the generic IPv4/IPv6 Extended Reachability Attributes sub-TLV defined in IETF RFC 7794, with similar semantics as in the Segment Routing ISIS extensions IETF draft. This sub-TLV permits to advertise prefix attributes independently from any application such as Segment Routing. Likely only the flags in the Extended Reachability Attributes sub-TLV will eventually remain. Since currently deployed Segment Routing implementations rely on the N- and R-flags in the Prefix-SID sub-TLV, there will be a transition period where both sets of flags will be used. The flags in the Extended Reachability Attributes sub-TLV are preferred in the case both sets of flags are received.

The R-flag is used in multi-level networks or when using redistribution. See the multi-level section 5.5.

V- and L-flags are currently always unset in IOS XR: Prefix Segments are always Global Segments with SIDs distributed in the form of SID indexes.

- The Algorithm field in the sub-TLV contains the identifier of the algorithm that the node used to compute the reachability of the prefix that the Prefix-SID is associated with. At the time of writing this book, only the well-known metric base SPF algorithm (0) is supported in ISIS. The Strict-SPF algorithm would be added here in the future.

- The SID/Index/Label field in the Prefix-SID sub-TLV contains the Prefix-SID index if V-flag and L-flag are unset, which is now the only supported combination in Cisco IOS XR.

The network topology of Figure 5-10 is used to illustrate the advertisement of Prefix-SIDs in ISIS. Segment Routing is enabled on both nodes and for both IPv4 and IPv6 address-families under IS-IS. The default multi-topology model is used. The configuration of Node1 is displayed in Example 5-14.

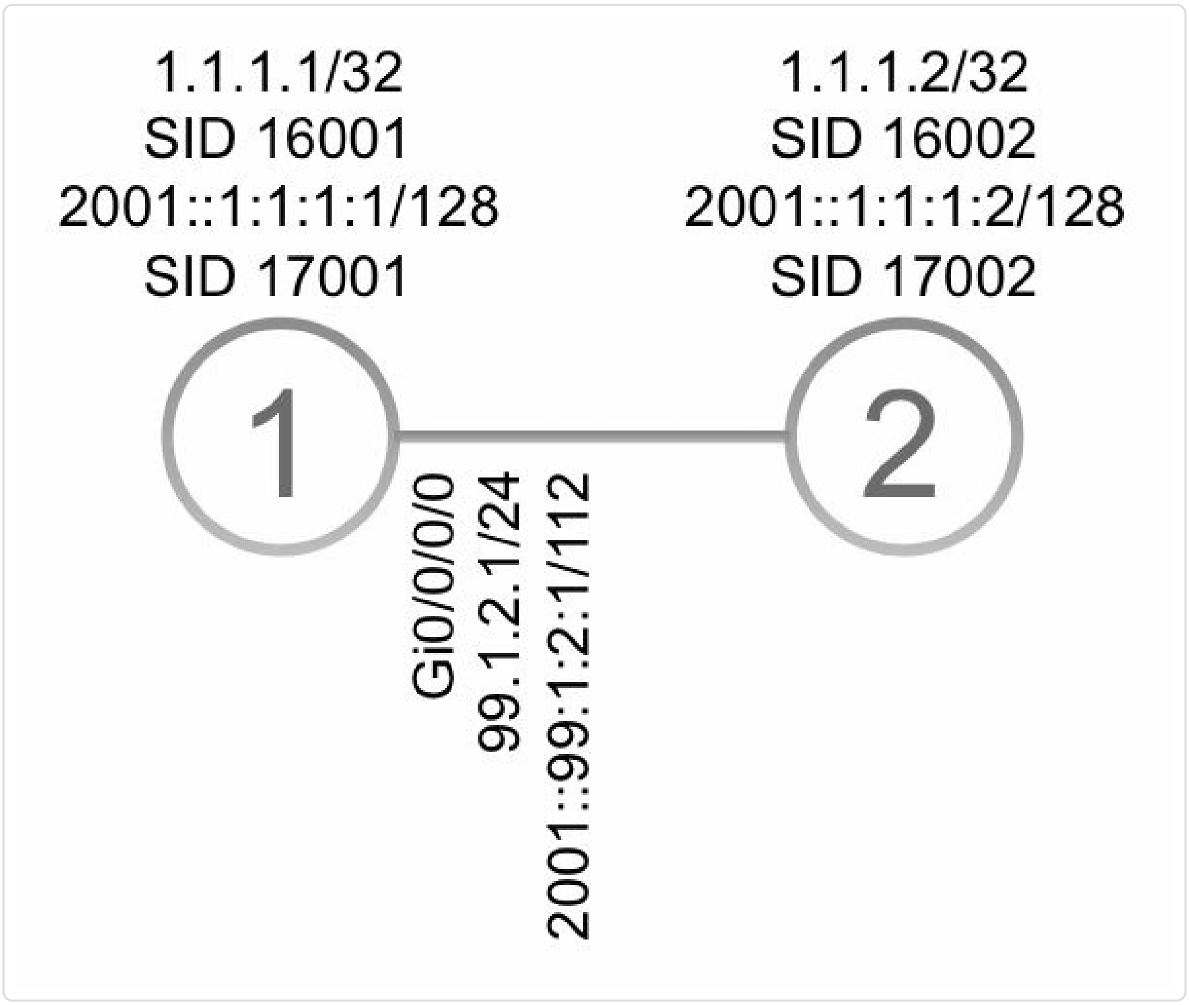


Figure 5-10: ISIS Prefix-SID – Network topology example

Example 5-14: ISIS Prefix-SID – configuration example Node1

```
interface Loopback0
```

```
  ipv4 address 1.1.1.1 255.255.255.255
  ipv6 address 2001::1:1:1:1/128
 !
 router isis 1
  !! default SRGB [16000-23999]
  address-family ipv4 unicast
   metric-style wide
   segment-routing mpls
  !
  address-family ipv6 unicast
   metric-style wide
   segment-routing mpls
  !
  interface Loopback0
   address-family ipv4 unicast
    prefix-sid absolute 16001
    !! Or: prefix-sid index 1
   !
   address-family ipv6 unicast
    prefix-sid absolute 17001
    !! Or: prefix-sid index 1001
  !
```

The output of `show isis database verbose` of the ISIS Link State PDU (LSP) advertised by Node1 in Example 5-15 shows the Loopback0 IPv4 and IPv6 prefixes advertised in their IP Reachability TLVs. The Prefix-SID sub-TLVs are attached to these IP Reachability TLVs.

The Loopback0 interface IPv4 address is 1.1.1.1/32 and the IPv4 Prefix-SID is configured as an absolute value 16001. In the Prefix-SID sub-TLV the Prefix-SID index 1 is advertised (16000 + 1 = 16001). The Algorithm field is set to 0, which is the default IGP metric shortest path algorithm. All Prefix-SID sub-TLV flags are unset, except the N-flag, which is set (`R:0 N:1 P:0 E:0 V:0 L:0`).

The Loopback0 interface IPv6 address is 2001::1:1:1:1/128 and the IPv6 Prefix-SID is configured as an absolute value 17001. In the Prefix-SID sub-TLV the Prefix-SID index 1001 is advertised (16000 + 1001 = 17001). Same as for the IPv4 Prefix-SID, the Algorithm field is set to 0 and all Prefix-SID sub-TLV flags are unset, except the N-flag.

Example 5-15: ISIS Prefix-SID advertisement example

```
RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-1
IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x0000039b   0xfc27
1079             0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6
Unicast                                     0/0/0
  Hostname:     xrvr-1
  IP Address:   1.1.1.1
  IPv6 Address: 2001::1:1:1:1
  Router Cap:   1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
<...>
  Metric: 0          IP-Extended 1.1.1.1/32
    Prefix-SID Index: 1, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
<...>
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
    Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
<...>
```

Example 5-16 and Example 5-17 show the MPLS forwarding entries on Node2 for the Prefix-SIDs 16001 and 17001 originated by Node1. Node2 is a penultimate hop node for these Prefix Segments hence it installs the Prefix-SID label with Pop outgoing label.

Example 5-16: IPv4 Prefix-SID MPLS forwarding entry

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 16001
Local  Outgoing    Prefix             Outgoing     Next
Hop          Bytes
Label  Label       or ID
Interface                      Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16001  Pop         SR Pfx (idx 1)     Gi0/0/0/0    99.1.2.1
      0
```

Example 5-17: IPv6 Prefix-SID MPLS forwarding entry

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 17001
Local  Outgoing    Prefix             Outgoing     Next
Hop          Bytes
Label  Label       or ID
Interface                      Switched
------ ----------- ------------------ ------------ ------------
--- ------------
17001  Pop         SR Pfx (idx 1001)  Gi0/0/0/0
fe80::f816:3eff:fe1e:55e1    \

 0
```

5.3.3.1 ISIS Prefix-SID with N-flag Unset

The network topology in Figure 5-11 illustrates the advertisement of regular Prefix-SIDs in ISIS. “Regular Prefix-SID” meaning: a Prefix-SID that is not a Node-SID. On both nodes, Segment Routing is enabled on both IPv4 and IPv6 address-families under IS-IS. The nodes advertise the indicated prefixes with Prefix-SIDs.

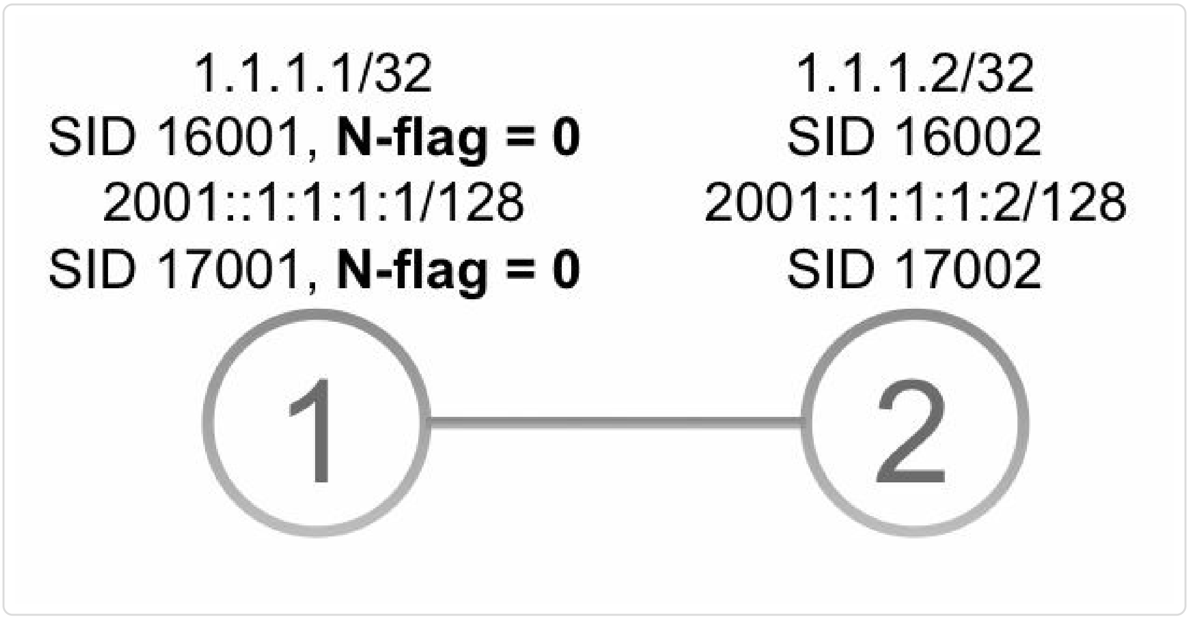


Figure 5-11: ISIS Prefix-SID – n-flag-clear network topology

Node1 uses the `n-flag-clear` keyword on the Prefix-SID configuration under interface Loopback0 for both address-families. See configuration of Node1 in Example 5-18.

Example 5-18: ISIS Prefix-SID – n-flag-clear configuration Node1

```
interface Loopback0
 ipv4 address 1.1.1.1 255.255.255.255
 ipv6 address 2001::1:1:1:1/128
!
router isis 1
 interface Loopback0
  address-family ipv4 unicast
   prefix-sid absolute 16001 n-flag-clear
  !
  address-family ipv6 unicast
   prefix-sid absolute 17001 n-flag-clear
```

The output of `show isis database verbose` in Example 5-19 shows that the N-flag is unset for both Prefix-SIDs in the ISIS advertisement of Node1.

Example 5-19: ISIS Prefix-SID – n-flag-clear advertisement

```
RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-1
<...>
  Metric: 0          IP-Extended 1.1.1.1/32
    Prefix-SID Index: 1, Algorithm:0, R:0 N:0 P:0 E:0 V:0 L:0
<...>
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
    Prefix-SID Index: 1001, Algorithm:0, R:0 N:0 P:0 E:0 V:0
L:0
<...>
```

The N-flag has no influence on the forwarding entry that is programmed for the Prefix-SID. The N-flag is more of concern for the control plane to identify prefixes that identify a node.

5.3.3.2 ISIS Prefix-SID with Explicit-null Flag

The network topology in Figure 5-12 is used to illustrate the IS-IS Prefix-SID advertisement requesting Explicit-null behavior from the penultimate hop for a Prefix Segment.

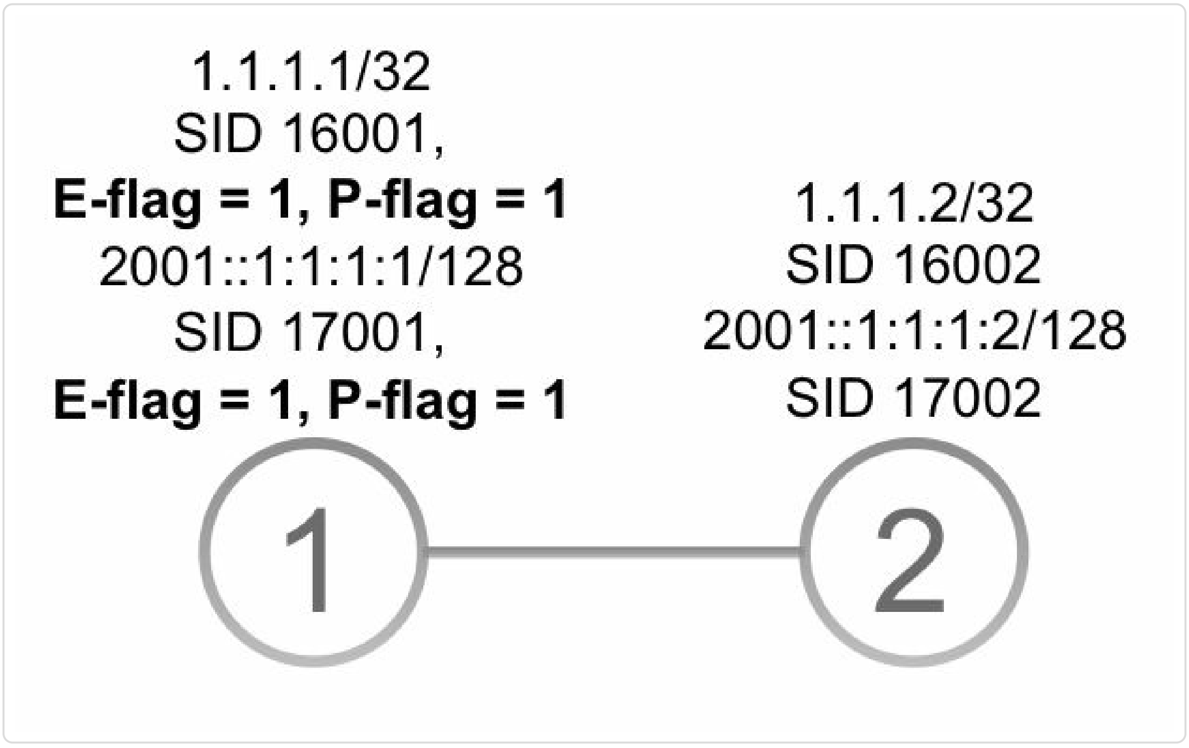


Figure 5-12: ISIS Prefix-SID Explicit-null – Network topology example

Node1 uses the `explicit-null` keyword on the Prefix-SID configuration under interface Loopback0 for both address-families. See configuration in Example 5-20.

Example 5-20: ISIS Prefix-SID Explicit-null – Configuration example

```
interface Loopback0
 ipv4 address 1.1.1.1 255.255.255.255
 ipv6 address 2001::1:1:1:1/128
!
router isis 1
 interface Loopback0
  address-family ipv4 unicast
   prefix-sid absolute 16001 explicit-null
  !
  address-family ipv6 unicast
   prefix-sid absolute 17001 explicit-null
```

The output of `show isis database verbose` shows that both P- and E-flag are set for both Prefix-SIDs. See lines 4 and 7 in Example 5-21. Since the `n-flag-clear` configuration option has not been configured the Prefix-SIDs are also Node-SIDs (N-flag set, `N:1`).

Example 5-21: ISIS Prefix-SID Explicit-null – Advertisement example

```
1 RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-1
2 <...>
3   Metric: 0           IP-Extended 1.1.1.1/32
4     Prefix-SID Index: 1, Algorithm:0, R:0 N:1 P:1 E:1 V:0 L:0
5 <...>
6   Metric: 0           MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
7     Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:1 E:1 V:0
L:0
8 <...>
```

The MPLS forwarding table collected on Node2 shows the outgoing explicit-null label for both Prefix-SID label entries. See output in Example 5-22 and Example 5-23. For the IPv4 Prefix-SID entry the IPv4 explicit-null label has been programmed (label value 0), for the IPv6 Prefix-SID the IPv6 explicit-null label has been programmed (label value 2).

Example 5-22: ISIS Prefix-SID Explicit-null – IPv4 MPLS forwarding example

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 16001
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16001  **Exp-Null-v4** SR Pfx (idx 1)     Gi0/0/0/0    99.1.2.1
      0
```

Example 5-23: ISIS Prefix-SID Explicit-null – IPv6 MPLS forwarding example

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 17001
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
pass:q[17001  *Exp-Null-v6* SR Pfx (idx 1001)  Gi0/0/0/0
fe80::f816:3eff:fe1e:55e1   \]

 0
```

5.3.4 Prefix-SID Advertisement in OSPFv2

The fact that a prefix identifies the node that originates this prefix is a property or attribute of that prefix; such prefix is a “node prefix”. This prefix property is independent of Segment Routing and other applications besides Segment Routing can make use of that prefix property. Therefore the N-flag can be distributed independently from the Segment Routing advertisements. This is already the case for OSPF: the OSPFv2 Extended Prefix TLV, which carried the prefix's N-flag, can be advertised independently from Segment Routing.

OSPFv2 advertises a loopback prefix inside an area as a stub-link in a Router LSA (Type 1 LSA). When this prefix is propagated to another area, a Summary LSA (Type 3 LSA) is used. These LSAs have a fixed-format, they are not extendable. None of these LSAs provide a way to advertise a SID with a prefix. Therefore OSPF uses an OSPFv2 Extended Prefix Opaque LSA defined in RFC 7684 to advertise a Prefix-SID for a prefix.

The OSPFv2 Extended Prefix Opaque LSA is of Opaque Type 7 and may use area flooding scope (i.e. sent as OSPF LSA type 10) or AS-wide flooding scope (i.e. sent as OSPF LSA type 11). The OSPFv2 Extended Prefix Opaque LSA format consists of the LSA header (see section 5.1.1.1 for more details) and a number of TLVs. The Opaque ID field in the header is an arbitrary number that is only used to differentiate multiple Opaque LSAs advertised by the same node.

The OSPFv2 Extended Prefix Opaque LSA format is displayed in Figure 5-13

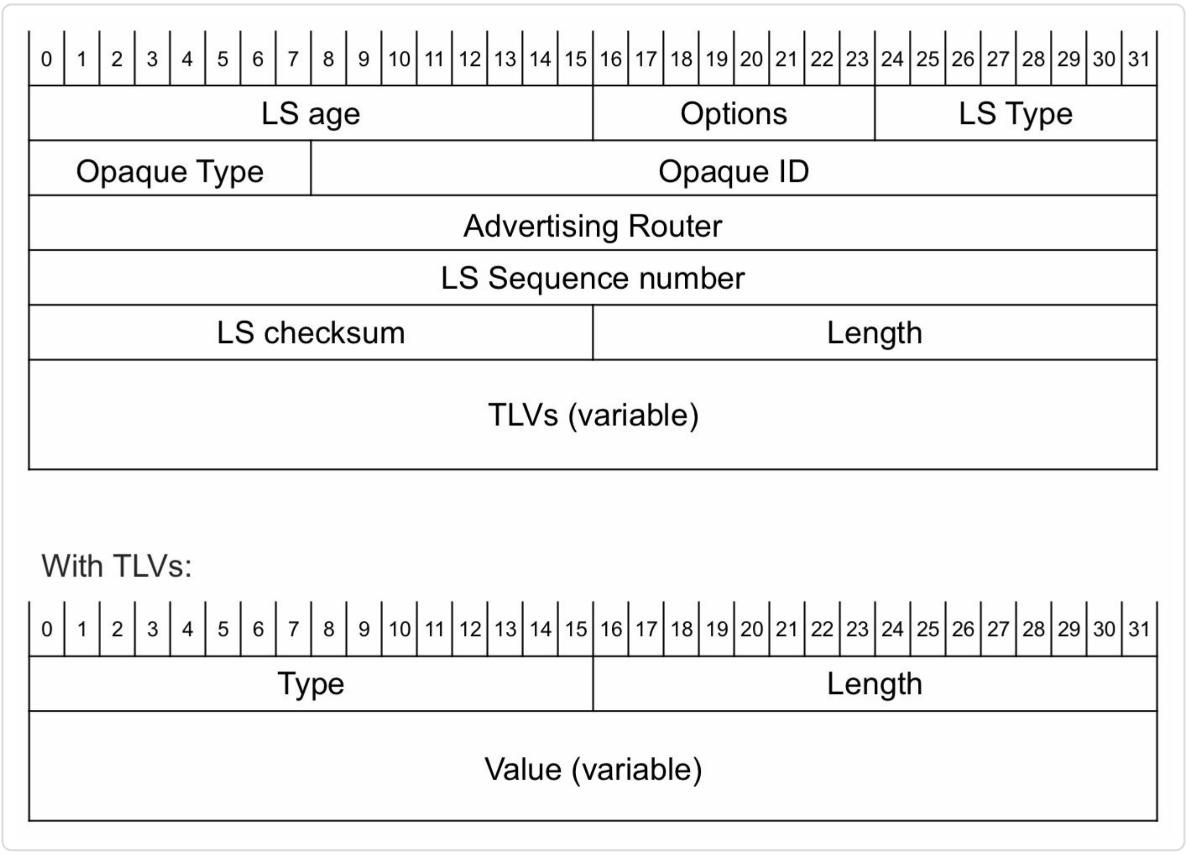


Figure 5-13: OSPFv2 Extended Prefix Opaque LSA format

OSPFv2 Extended Prefix TLV has been defined in IETF RFC 7684 to be carried in the OSPFv2 Extended Prefix Opaque LSA. The OSPFv2 Extended Prefix Opaque LSA and TLV provide a generic way to distribute prefix attributes in the OSPF network, their usage is not limited to Segment Routing.

The OSPFv2 Extended Prefix TLV has the format as shown in Figure 5-14.

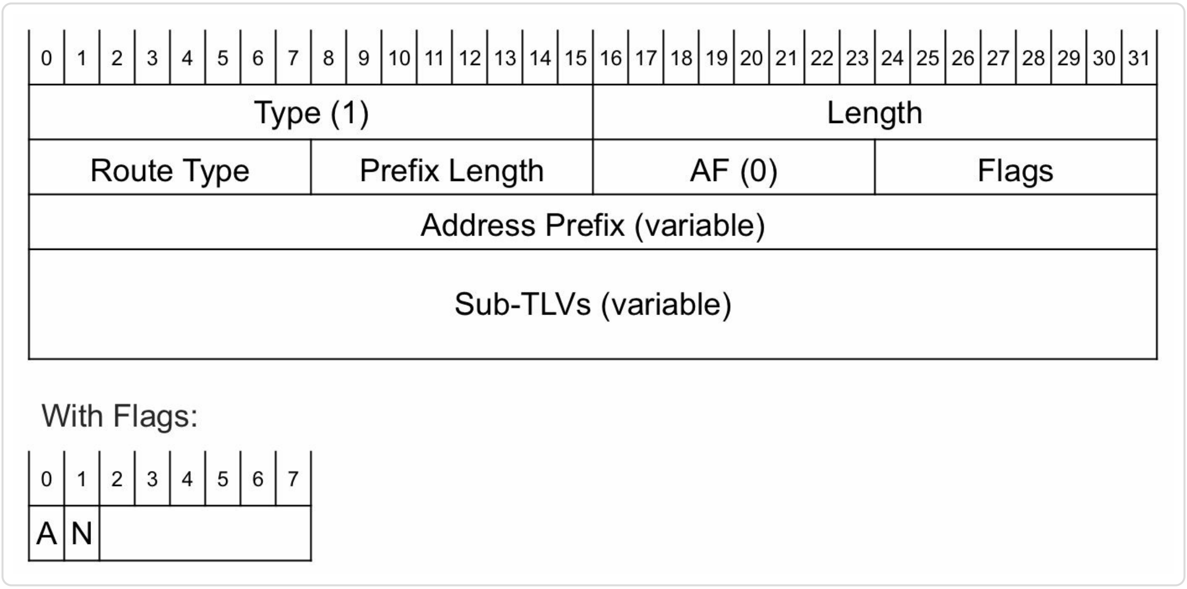


Figure 5-14: OSPFv2 Extended Prefix TLV format

This TLV contains the necessary fields to uniquely identify a prefix in the OSPF domain.

The Route Type field in the TLV specifies the type of the OSPFv2 prefix that this TLV applies to. The field value correspond to the OSPFv2 LSA types:

- 0 - Unspecified
- 1 - Intra-Area
- 3 - Inter-Area
- 5 - Autonomous System (AS) External
- 7 - Not-So-Stubby Area (NSSA) External

If the Route Type is 0 (Unspecified), then the information inside the OSPFv2 Extended Prefix TLV applies to the prefix regardless of the prefix's route type. This is useful when prefix-specific attributes are advertised by an external entity that is not aware of the route type associated with the prefix.

The Address Prefix field and Prefix Length field contain the prefix that this TLV applies to and its prefix length.

The Address Family (AF) field in the TLV specifies the address-family of the prefix. This field is included for future extensions, currently only address-family IPv4 unicast (0) is supported.

The Flags field in the TLV contains these flags:

- A-Flag (Attach): Set by an Area Border Router (ABR) for an inter-area prefix that is locally connected or attached to the ABR, but in another connected area that the area in which this LSA has been advertised. In other words, this inter-area prefix is connected to the ABR and propagated between another connected area and this area. Cisco IOS XR: currently always unset.
- N-Flag (Node): Set when the prefix identifies the advertising router. This flag is preserved when the OSPFv2 Extended Prefix Opaque LSA is propagated between areas. Cisco IOS XR: set by default.

The OSPFv2 Extended Prefix TLV uniquely identifies the prefix to which the attributes apply. And it also carries two prefix attributes, two properties of the prefix: the A-flag and N-flag. Other attributes are carried in the sub-TLVs included in this TLV. One of the possible sub-TLVs is the Prefix-SID sub-TLV.

A Prefix-SID is advertised in a Prefix-SID sub-TLV, which is a sub-TLV of the OSPF Extended Prefix TLV. The Prefix-SID sub-TLV is defined in

IETF draft-ietf-ospf-segment-routing-extensions and has the format displayed in Figure 5-15.

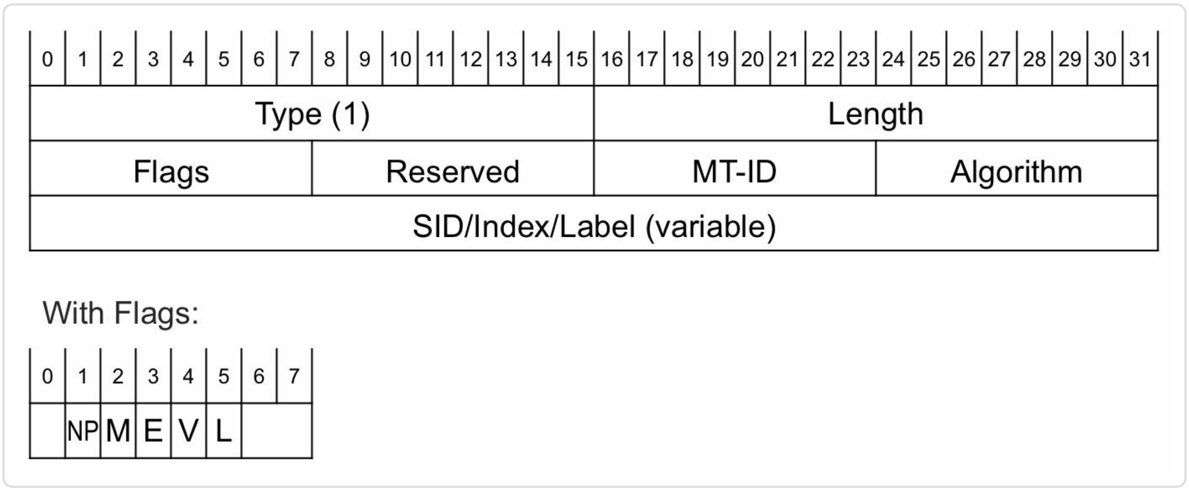


Figure 5-15: OSPFv2 Prefix-SID sub-TLV

- The Flags field in the TLV contains the following flags:
  - NP: no-PHP or PHP-off: if set, then the penultimate hop must not pop the prefix-SID before forwarding the packet – default in Cisco IOS XR: unset, i.e. PHP is on by default.
  - M: Mapping Server: set if the SID is advertised from a Mapping Server – default in Cisco IOS XR: unset
  - E: Explicit-Null: if set, then the penultimate hop must replace the prefix-SID label with the Explicit-Null label before forwarding the packet – default in Cisco IOS XR: unset
  - V: Value: set if prefix-SID carries an absolute value, unset if the Prefix-SID carries an index – Cisco IOS XR: always unset
  - L: Local/Global: set if prefix-SID has local significance, unset if prefix-SID has global significance – Cisco IOS XR: always unset
- MT-ID field: Multi-Topology ID (as defined in IETF RFC 4915). – Cisco IOS XR: always 0

- Algorithm field: the algorithm the Prefix-SID is associated with. At the time of writing this book, the well-known metric base SPF(0) and Strict SPF (1) are supported – IOS XR: always 0 or 1.
- SID/Index/Label field contains the Prefix-SID index if V-flag and L-flag are unset, the only supported combination in IOS XR.

The network topology in Figure 5-16 is used to illustrate the advertisement of Prefix-SIDs in OSPFv2. Segment Routing is enabled on both nodes in this topology and each node advertises a Prefix-SID. The configuration of Node1 is shown in Example 5-24.

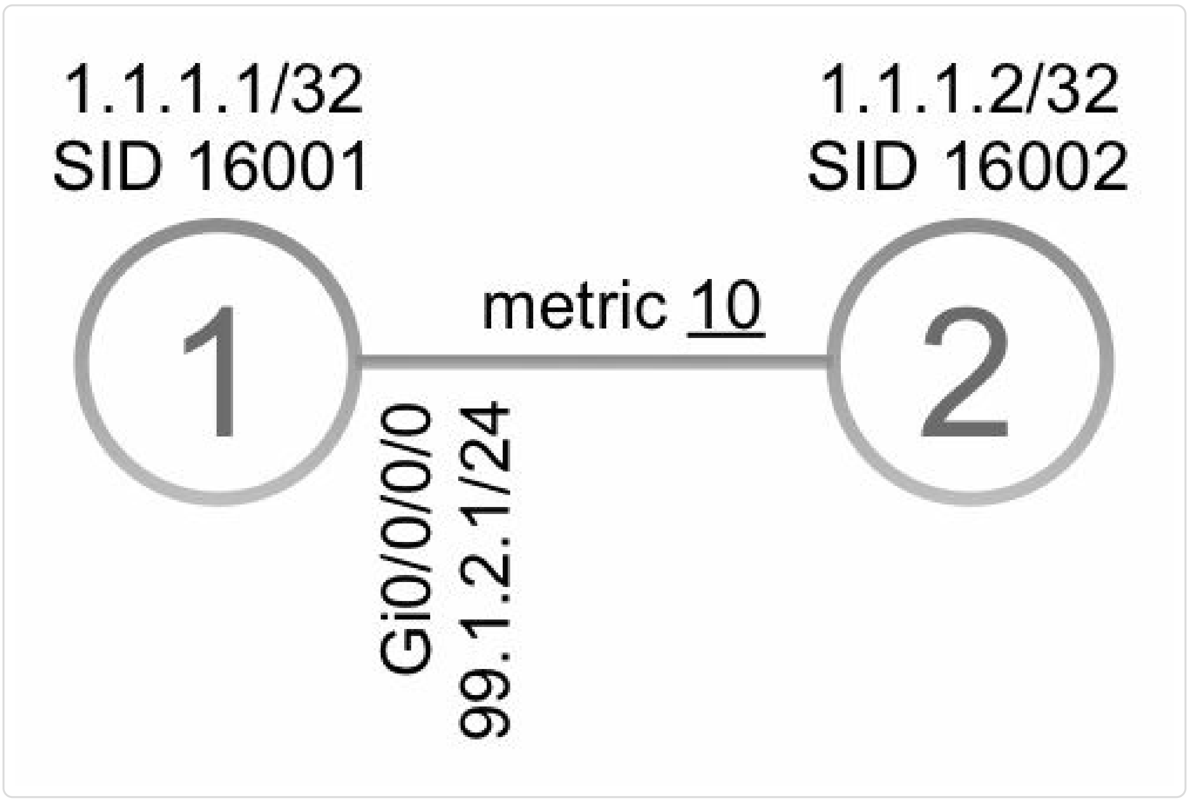


Figure 5-16: OSPFv2 Prefix-SID – Network topology example

Example 5-24: OSPFv2 Prefix-SID – Configuration example Node1

```
interface Loopback0
 description Loopback
 ipv4 address 1.1.1.1 255.255.255.255
!
```

```
router ospf 1
 area 0
  interface Loopback0
   passive enable
   prefix-sid absolute 16001
  !
 !
!
```

The output of `show ospf database router` of the Router LSA advertised by Node1 in Example 5-25 shows the Loopback0 IPv4 prefix 1.1.1.1/32 advertised as stub network. The other stub network is the interface network prefix of the link to Node2. This Router LSA contains no Segment Routing information.

Example 5-25: OSPFv2 Prefix-SID – Router LSA

```
RP/0/0/CPU0:xrvr-1#show ospf database router self-originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 0)

  LS age: 4
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.1.1.1
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000216
  Checksum: 0x9a60
  Length: 60
  Area Border Router
   Number of Links: 3

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 1.1.1.1
     (Link Data) Network Mask: 255.255.255.255
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 1.1.1.1
     (Link Data) Router Interface address: 99.1.2.1
      Number of TOS metrics: 0
       TOS 0 Metrics: 10

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 99.1.2.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 10
```

There is no direct link between the Router LSA and the Extended Prefix LSA(s). The Prefix-SID is linked to the prefix by identifying the prefix in the Extended Prefix LSA. Note that this is only to reference the prefix, it is not used to advertise prefix reachability. The command `show ospf database opaque-area 1.1.1.1/32` displays the Extended Prefix LSA that applies to prefix 1.1.1.1/32. See Example 5-26. The LSA is a type 10 LSA with Opaque Type 7. The Opaque ID for this LSA is 1. This Opaque ID field is only used to differentiate multiple Opaque Type 7 LSAs advertised by the same node; it has no other significance. The Opaque Type and Opaque ID are combined and presented as a dotted decimal Link State ID in the command output: “7.0.0.1” in the example. It is an identifier, do not confuse it with an IPv4 address.

Example 5-26: OSPFv2 Prefix-SID – Extended Prefix LSA

```
RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 1.1.1.1/32

            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 0)

  LS age: 1802
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
```

```
    Link State ID: 7.0.0.1
    Opaque Type: 7
    Opaque ID: 1
    Advertising Router: 1.1.1.1
    LS Seq Number: 80000214
    Checksum: 0xf983
    Length: 44

      Extended Prefix TLV: Length: 20
        Route-type: 1                   !! intra-area prefix
        AF        : 0                   !! IPv4 unicast
        Flags     : 0x40                !! A:0, N:1
        Prefix    : 1.1.1.1/32

        SID sub-TLV: Length: 8
          Flags     : 0x0               !! NP:0, M:0, E:0, V:0,
  L:0
          MTID      : 0                 !! default topology
          Algo      : 0                 !! regular SPF
          SID Index : 1
```

The prefix that the Extended Prefix TLV in this example applies to is a prefix of Route-type 1, which is an intra-area prefix. It is a prefix of address-family (AF) 0, which is IPv4 unicast. The prefix is 1.1.1.1/32.

The Extended Prefix TLV itself contains two prefix attribute flags. One prefix flag is set: the N-flag (0x40). Hence this prefix is a Node prefix; it identifies the node that originates the prefix. The A-flag is not set; this flag is only relevant for inter-area prefixes.

The Prefix-SID sub-TLV (named `SID sub-TLV` in the output) specifies the Prefix-SID associated with the prefix 1.1.1.1/32. None of the Prefix-SID flags is set (`Flags: 0x0`), which implies that default PHP behavior is requested from the penultimate hop. The prefix resides in the default topology with Multi-Topology ID (MTID) 0. The Prefix-SID is associated with the algorithm 0 (`Algo: 0`), which is the regular IGP metric SPF. The SID index is 1.

This section only shows an example of a Prefix-SID for an intra-area prefix. Other prefix types are illustrated in the multi-area section 5.5 and in the redistribution section 5.6.

Example 5-27 shows the MPLS forwarding entries on Node2 for the Prefix-SID 16001 originated by Node1. Node2 is a penultimate hop node for these Prefix Segments and it installs the Prefix-SID label with Pop outgoing label.

Example 5-27: OSPFv2 Prefix-SID – MPLS forwarding entry

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 16001
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID              Interface
   Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16001  Pop         SR Pfx (idx 1)     Gi0/0/0/0    99.1.2.1
      0
```

5.3.4.1 OSPF Prefix-SID with N-flag Unset

The network topology inFigure 5-17 illustrates the advertisement of regular, non-Node Prefix-SIDs in OSPF.

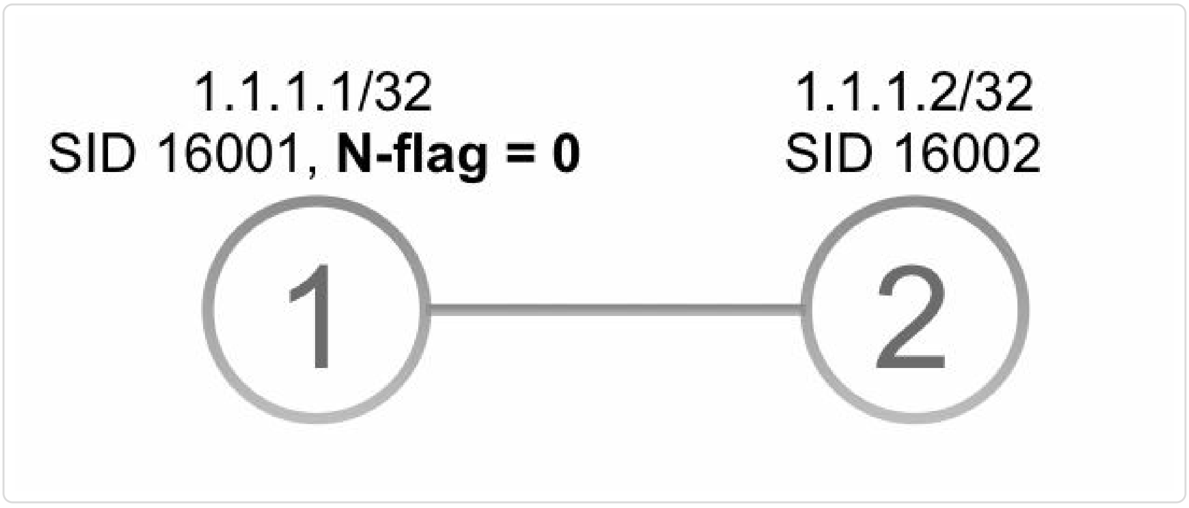


Figure 5-17: OSPFv2 Prefix-SID – n-flag-clear network topology

Node1 uses the `n-flag-clear` keyword on the Prefix-SID configuration under interface Loopback0. See configuration of Node1 in Example 5-28.

Example 5-28: ISIS Prefix-SID – n-flag-clear configuration Node1

```
interface Loopback0
 description Loopback
 ipv4 address 1.1.1.1 255.255.255.255
!
router ospf 1
 area 0
  interface Loopback0
   passive enable
   prefix-sid absolute 16001 n-flag-clear
  !
 !
!
```

The output of `show ospf database opaque-area 1.1.1.1/32` in Example 5-29 shows that the N-flag is unset (`Flags: 0x0`) in the Prefix-SID advertisement of prefix 1.1.1.1/32 (line 22).

Example 5-29: ISIS Prefix-SID – n-flag-clear advertisement

```
RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 1.1.1.1/32


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 0)

  LS age: 4
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.1
  Opaque Type: 7
  Opaque ID: 1
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000001
  Checksum: 0x6373
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 1                      !! intra-area prefix
      AF        : 0                      !! IPv4 unicast
      Flags     : 0x0                    !! A:0, N:0
      Prefix    : 1.1.1.1/32

      SID sub-TLV: Length: 8
        Flags     : 0x0                  !! NP:0, M:0, E:0, V:0, L:0
        MTID      : 0                    !! default topology
        Algo      : 0                    !! regular SPF
        SID Index : 1

```

5.3.4.2 OSPF Prefix-SID with Explicit-null Flag

The network topology in Figure 5-18 is used to illustrate the OSPF advertisement requesting Explicit-null behavior from the penultimate hop for a Prefix Segment.

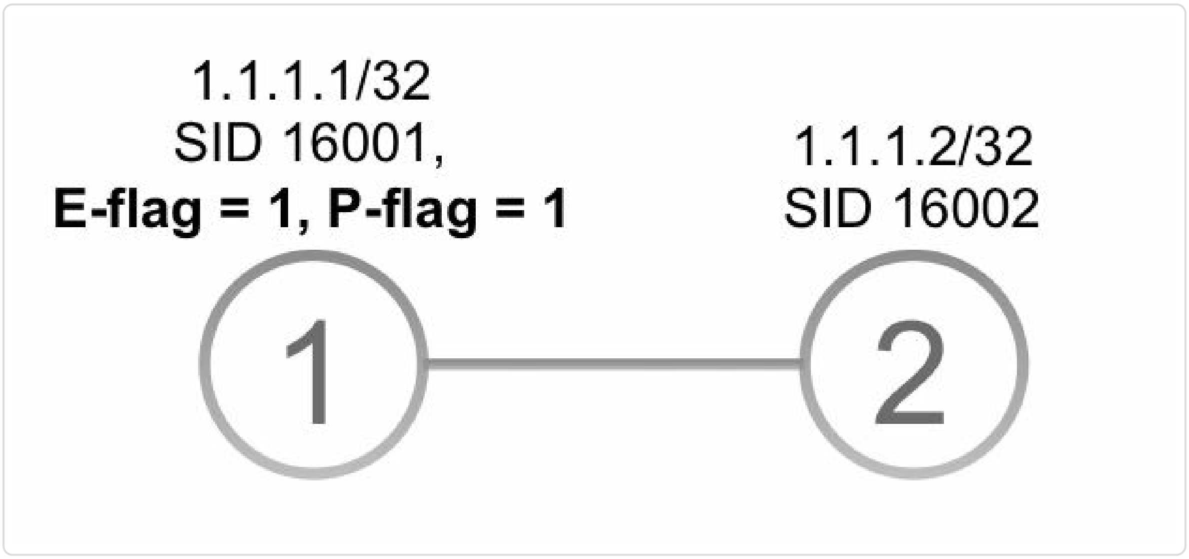


Figure 5-18: OSPFv2 Prefix-SID Explicit-null – Network topology example

Node1 uses the `explicit-null` keyword on the Prefix-SID configuration under interface Loopback0. See configuration of Node1 in Example 5-30.

Example 5-30: OSPFv2 Prefix-SID Explicit-null – Configuration example

```
interface Loopback0
 ipv4 address 1.1.1.1 255.255.255.255
!
router ospf 1
 area0
  interface Loopback0
   prefix-sid absolute 16001 explicit-null
```

The output of `show ospf database opaque-area` for the OSPFv2 Extended Prefix Opaque LSA associated with prefix 1.1.1.1/32 shows that both NP-flag (0x40) and E-flag (0x10) are set in the Prefix-SID sub-TLV. The other flags are unset. See line 26 in Example 5-31. Note that bit 0 in the Flags field is unused, that must be taken into account when interpreting the hexadecimal representation of the Flags field. With these flags, Node1 requests the penultimate hop node to swap the Prefix-SID label with explicit-null label.

Example 5-31: OSPFv2 Prefix-SID Explicit-null – Advertisement example

```
RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 1.1.1.1/32


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 0)

  LS age: 1802
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.1
  Opaque Type: 7
  Opaque ID: 1
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000214
  Checksum: 0xf983
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 1
      AF        : 0
      Flags     : 0x40                !! A:0, N:1
      Prefix    : 1.1.1.1/32

      SID sub-TLV: Length: 8
        Flags     : 0x50              !! RESV, NP:1, M:0, E:1, V:0, L:0
        MTID      : 0
        Algo      : 0
        SID Index : 1

```

The MPLS forwarding table collected on Node2 shows the outgoing explicit-null label for the Prefix-SID label entry. See the output in Example 5-32.

Example 5-32: OSPFv2 Prefix-SID Explicit-null – MPLS forwarding example

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 16001
Local  Outgoing    Prefix             Outgoing     Next
Hop        Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16001  Exp-Null-v4 SR Pfx (idx 1)     Gi0/0/0/0
99.1.2.1        0
```

5.4 Adjacency-SID

The description of Adjacency-SID and its different variants have been previously covered in chapter 2, “Fundamentals of Segment Routing”. In this section, we will focus on how Adjacency-SIDs are allocated by IGPs and the IGP mechanisms related to their advertisement and programming of corresponding forwarding entries.

5.4.1 Signaling Adjacency-SID in IGP Advertisements

Link State IGPs today already signal the network topology in a distributed manner in terms of nodes and the links/adjacencies between them. It is natural to extend the link attributes that are signaled by each node to also include the Segment Routing information viz. Adjacency Segments.

We have discussed the IGP Adjacency Segments previously in chapter 2, “Fundamentals of Segment Routing” and that they are typically Local Segments. The current Cisco IOS-XR Segment Routing MPLS data plane implementation only uses *Local* IGP Adjacency Segments and further references in this book to “IGP Adjacency Segments” assume Local Segments.

IGP Adjacency-SIDs are automatically allocated from the dynamic label range for each adjacency. This dynamic label range is outside of the SRGB. Since Adjacency Segments are Local Segments, Adjacency-SIDs are advertised in the IGP as absolute MPLS label values.

Although each node in the domain receives the advertised Adjacency-SIDs from all other nodes, only the node originating the Adjacency-SID can correctly interpret it and only this node installs its own Adjacency-SIDs in the forwarding table. We will see further in the chapter 9, “Topology Independent LFA (TI-LFA)” where for the TI-LFA feature, the remote nodes can utilize the Adjacency-SIDs that they learn from other nodes to direct traffic on repair paths.

An IGP adjacency can have multiple Adjacency-SIDs associated with it. Cisco IOS XR implementation automatically allocates two Adjacency-SIDs for each adjacency: one which would be always unprotected (i.e. not have a backup path in event of failure) and one that is protection-eligible (i.e. will have a backup path, if available, that is programmed in the forwarding). The TI-LFA feature that we will describe further in chapter 9, provides automatic protection for these Adjacency-SID label forwarding entries against failures, by using fast-reroute (FRR) mechanisms to activate repair paths in the forwarding plane .

The unprotected Adjacency-SID can be used by applications where a local repair kind of traffic protection is not desired at intermediate points and traffic to be rather dropped when a failure occurs – perhaps to trigger an end-to-end backup path from the source itself. The Adjacency-SID that is protection-eligible can be used by applications that need a local repair kind of protection and intermediate points. These concepts are the building blocks for SR Traffic Engineering, which we shall go into more details in the next part of this book.

As of writing of this book, the implementations of the two other types of Adjacency-SIDs, namely L2 Adjacency-SID and Group Adjacency SID, are not yet released in the Cisco IOS XR software and we will perhaps cover them in detail in a future revision of this book.

5.4.2 Adjacency-SID Allocation

An IGP Adjacency Segment is a segment that is associated with a unidirectional adjacency. The instruction of an IGP Adjacency Segment is *steer traffic out of the link of the adjacency associated with this segment*. The traffic on this Adjacency Segment is steered on the link regardless of shortest path routing.

As an Adjacency Segment is a Local Segment, each node locally allocates its Adjacency Segments' local labels from the dynamic label range, outside of the SRGB, and advertises them as absolute label values associated with each adjacency.

ISIS allocates and advertises Adjacency-SIDs for each adjacency that is in "Up" state.

OSPF allocates and advertises Adjacency-SIDs for each adjacency that is in "2-way" state or above. If the adjacency goes below "2-way" state then OSPF withdraws the Adjacency-SIDs. As explained above, IS-IS and OSPF allocate two Adjacency-SIDs per adjacency: one that is protection-eligible and one that is not protected. The Adjacency-SIDs are advertised with the B-flag (Backup) set for the ones that are eligble for protection and with the B-flag unset for the unprotected ones. This B-flag is used to indicate each Adjacency-SID's protection capability to the other nodes in the network. Note, however, that an Adjacency-SID with B-Flag set does not necessarily indicated that it always has protection available (i.e. a repair path programmed in forwarding) at the originating node; though typically in most deployments with properly setup Segment Routing and TI-LFA the repair path would be available.

IS-IS allocates different Adjacency-SIDs for Level-1 and Level-2 adjacencies between the same pair of L1L2 neighbors. It also allocates different Adjacency-SIDs for IPv4 and IPv6 address-families on the same adjacency between the same neighbors.

OSPF allocates the same adjacency-SID in all the areas of a multi-area adjacency even if they are advertised individually along with the same link in each of the areas.

OSPF Virtual Links and Sham link adjacencies do not have an Adjacency-SID.

TE forwarding adjacencies are also a special type of links as seen by the IGPs but there is no hello adjacency between them and no Adjacency-SIDs are allocated to them. We will see in the next part of this book that deals with SR TE that there are other mechanisms like Binding-SID that allow directing SR traffic into a TE Policy path at a given node. Hence, an IGP Adjacency-SID is normally not required and not allocated in the Cisco IOS XR implementation for such adjacencies.

5.4.3 LAN Adjacency-SID

Although point-to-point adjacencies are more prevalent in networks, Segment Routing can also be used over a broadcast Local Area Network (LAN) such as an Ethernet LAN that are present in some network designs. This introductory section uses the term “LAN Adjacency-SID” to indicate an Adjacency-SID associated with an adjacency on a LAN interface. This deviates somewhat from the IGP (IS-IS or OSPF) specific sections that follow after this, where this term indicates specific types of (sub-)TLVs.

5.4.3.1 IGP LAN Adjacencies Reminder

Multiple devices on a LAN segment share a common broadcast medium; they can all reach each other over that medium. It would not be efficient to establish a full-mesh of point-to-point adjacencies from each node to each other node on the LAN. On a LAN segment with *n* nodes, there would be *n**(*n*-1)/2 adjacencies to maintain and a lot of unneeded flooding would occur on the LAN segment. To scale the number of adjacencies and the amount of flooding on a LAN, the concept of a “pseudonode” (called “designated router” or DR in OSPF) is used. Although the implementation and terminology is different between IS-IS and OSPF, both protocols use this concept.

A pseudonode is a virtual node that models or represents the broadcast medium, the “network”. Each node on the LAN advertises its “point-to-point” adjacency with this virtual pseudonode. The nodes on the LAN segment elect the node that takes the role of the pseudonode. The node that takes the role of the pseudonode advertises the adjacencies of the pseudonode with all nodes on the LAN on behalf of the pseudonode. See Figure 5-19. An elected Designated Intermediate System (DIS) plays the pseudonode role in IS-IS. An elected Designated Router (DR) plays that role in OSPF.

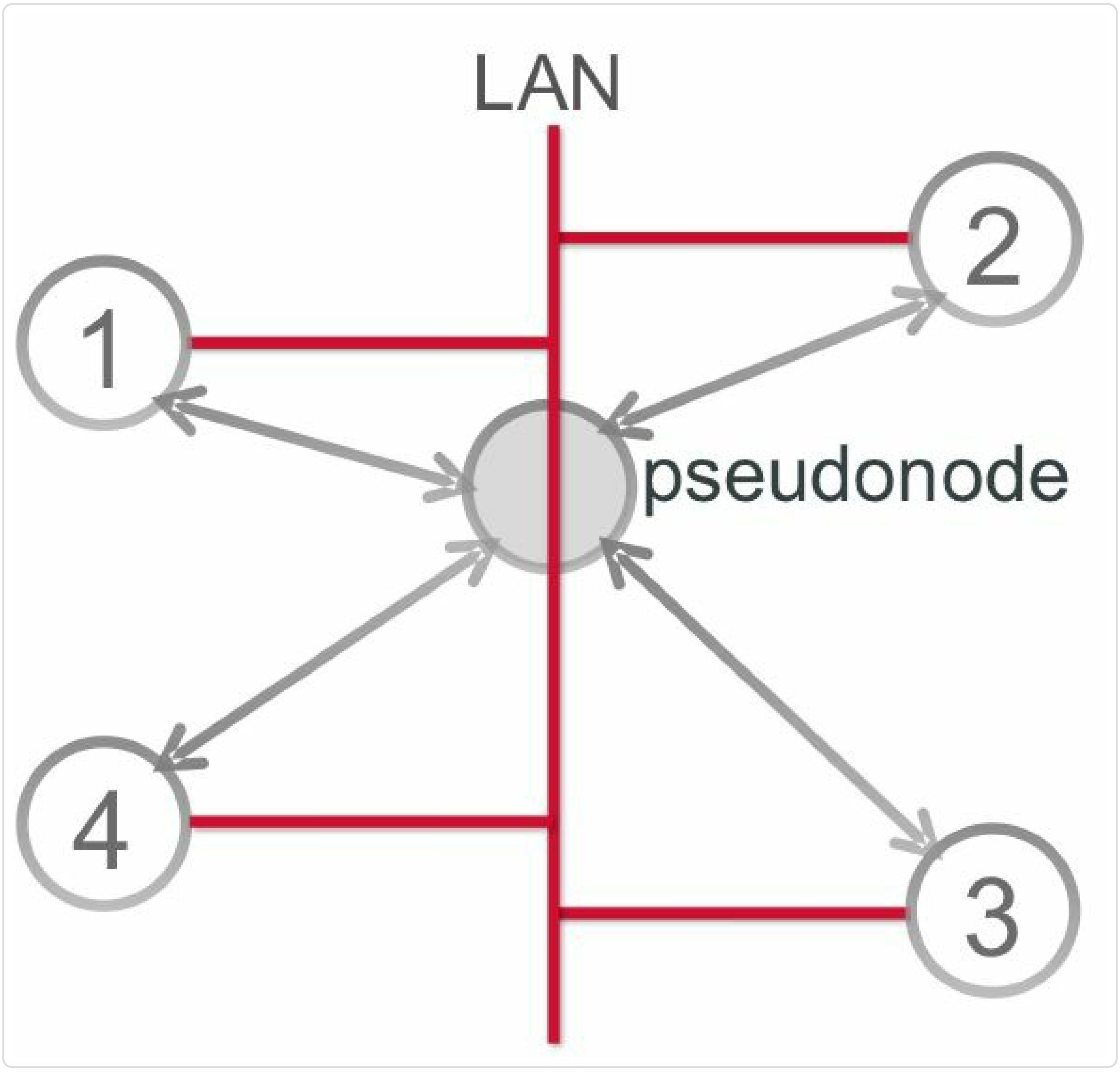


Figure 5-19: LAN with pseudonode

5.4.3.2 LAN Adjacency Segments

Even though a node on a LAN only advertises its adjacency to the LAN pseudonode, Segment Routing requires the capability to steer traffic directly to each of the other nodes on the LAN, not via a virtual pseudonode in the middle. Therefore each node on the LAN allocates and advertises a LAN Adjacency-SID towards each of the other nodes on the LAN.

Figure 5-20 illustrates this. Node1, Node2, Node3, and Node4 each advertises its adjacency to the pseudonode that models the LAN. Each node also allocates a LAN Adjacency-SID for each other node on the LAN. These LAN Adjacency-SIDs are advertised by attaching them to the node's adjacency to the pseudonode.

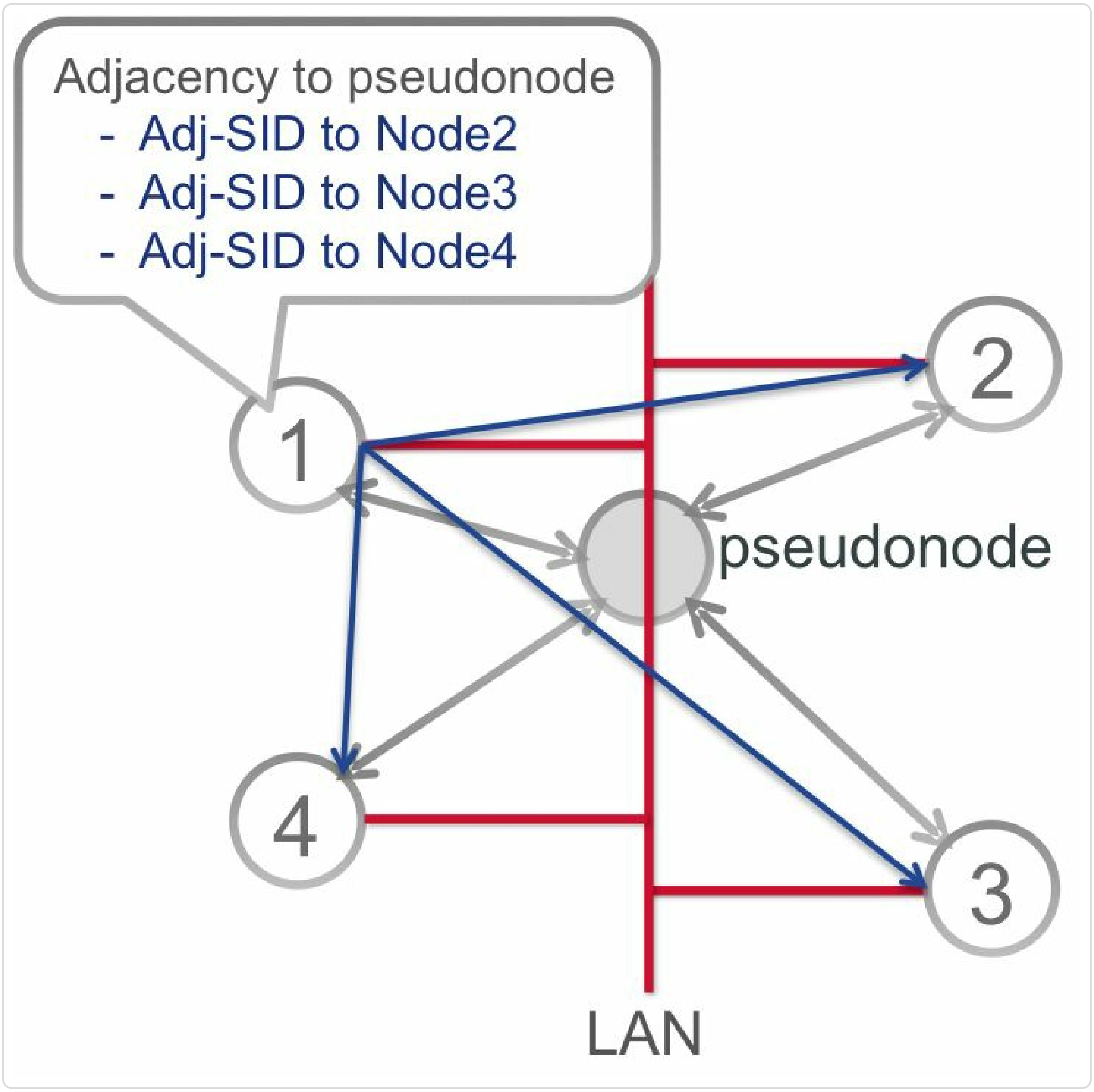


Figure 5-20: LAN with Adjacency-SIDs

For example in the network topology of Figure 5-20 Node1 advertises a single adjacency, the adjacency to the pseudonode. Node1 allocates LAN

Adjacency-SIDs to each of the other nodes on the LAN (Node2, Node3, and Node4). These Adjacency-SIDs allow steering traffic from Node1 to any of the other nodes on the LAN. Node1 advertises these LAN Adjacency-SIDs by attaching them to Node1's adjacency with the pseudonode.

Again, the term "LAN Adjacency-SID" in this introductory section is used as a generic term to indicate an Adjacency-SID associated with an adjacency on a LAN interface. It does not indicate how such adjacency is actually advertised in the IGP.

The allocation and advertisement of both Adjacency-SIDs and LAN Adjacency-SIDs together along with the links would become clearer further in this section as we look at how it is handled by ISIS and OSPF.

HIGHLIGHT

When using Ethernet links with IGPs, it is more efficient (not just for Segment Routing but also normal IGP operations) to always configure them as type point-to-point unless it connects to a real LAN segment with multiple routers on it.

5.4.4 Adjacency-SID Persistency

We have seen that the Prefix-SID are persistent in nature as they are Global Segments – i.e. their values will not change once provisioned in the network. On the other hand, Adjacency-SIDs are Local Segments (i.e. MPLS labels) that are allocated from the dynamic label pool. It may so happen that if an IGP adjacency were to go down (say due to a link flap), then when it comes up again, the dynamic label that its now allocated for the same Adjacency-SID is a different value than before. This would result in the possibility that the Adjacency-SID values keep changing and churning in the network. We will see in the next part of this book that Adjacency-SIDs can be used for by SR TE applications. Changes in their values on normally expected events like link flaps, can cause a lot of churn in updating the SR TE policies and paths used in the network, which is very inefficient. It is therefore highly desirable that the Adjacency-SIDs be persistent across most normal events in the network that affect the IGP adjacencies.

The Cisco IOS-XR implementation hence ensures that the allocated Adjacency-SID labels are persistent. When the adjacency fails and recovers again within a reasonable time (30 minutes), then the same label value as used before the failure will be allocated for the Adjacency-SID after recovering from that failure. This is ensured between the IGPs and the Label Switching Database (LSD), which manages the label space in the router.

If an adjacency with an Adjacency-SID goes down, IGP notifies LSD that the Adjacency-SID label is no longer in use. IGP "frees" the label. There is a difference between protection-eligible and unprotected Adjacency-SIDs when this notification happens, see later in this section. After LSD gets the notification from IGP, LSD does not immediately free the label and re-use that label for something else, but holds on to it for about 30 minutes..

If during that time period, LSD gets a request for a label for the same Adjacency-SID, then LSD allocates that same old label to the requesting IGP.

If LSD does not get a request for this Adjacency-SID label within the 30 minutes, then LSD makes the label available to *any* requestor. The label is *recycled*.

This persistency mechanism ensures that an adjacency that temporarily flaps will get the same Adjacency-SID label after it is restored within 30 minutes. This label persistency mechanism does not provide persistency for a full chassis reload, such as a power cycle.

If an adjacency goes down then the reaction of the IGP depends on the type of Adjacency Segment. For the unprotected Adjacency-SID, IGP immediately removes it since it is not protected and hence unusable after the adjacency failed. For the protection-eligible Adjacency-SID it is different. If it is protected, then traffic that is steered on this Adjacency-SID is still forwarded using the backup path of the Adjacency-SID (we will look at this concept of Adjacency Protection using TI-LFA in more detail in chapter 9). Therefore the IGP holds on to the protected Adjacency-SID for some time (5 minutes at the time of writing this book, but a longer delay period is under consideration) after the adjacency failed. This delay period allows time for any application that is using the protected Adjacency-SID to reroute the traffic to another path, avoiding this failed Adjacency Segment. We will discuss some of the SR TE applications and use-cases in the next part of this book. After this delay period, the IGP deletes the label entry and the 30 minute label preservation mechanism, as described above, comes into play.

The network topology in Figure 5-21 is used to illustrate this persistency behavior. In this network topology, one of the two links is shut down and kept down for some time. Then the link is restored and the Adjacency-SID label allocation will re-allocate the same labels to the Adjacency-SIDs.

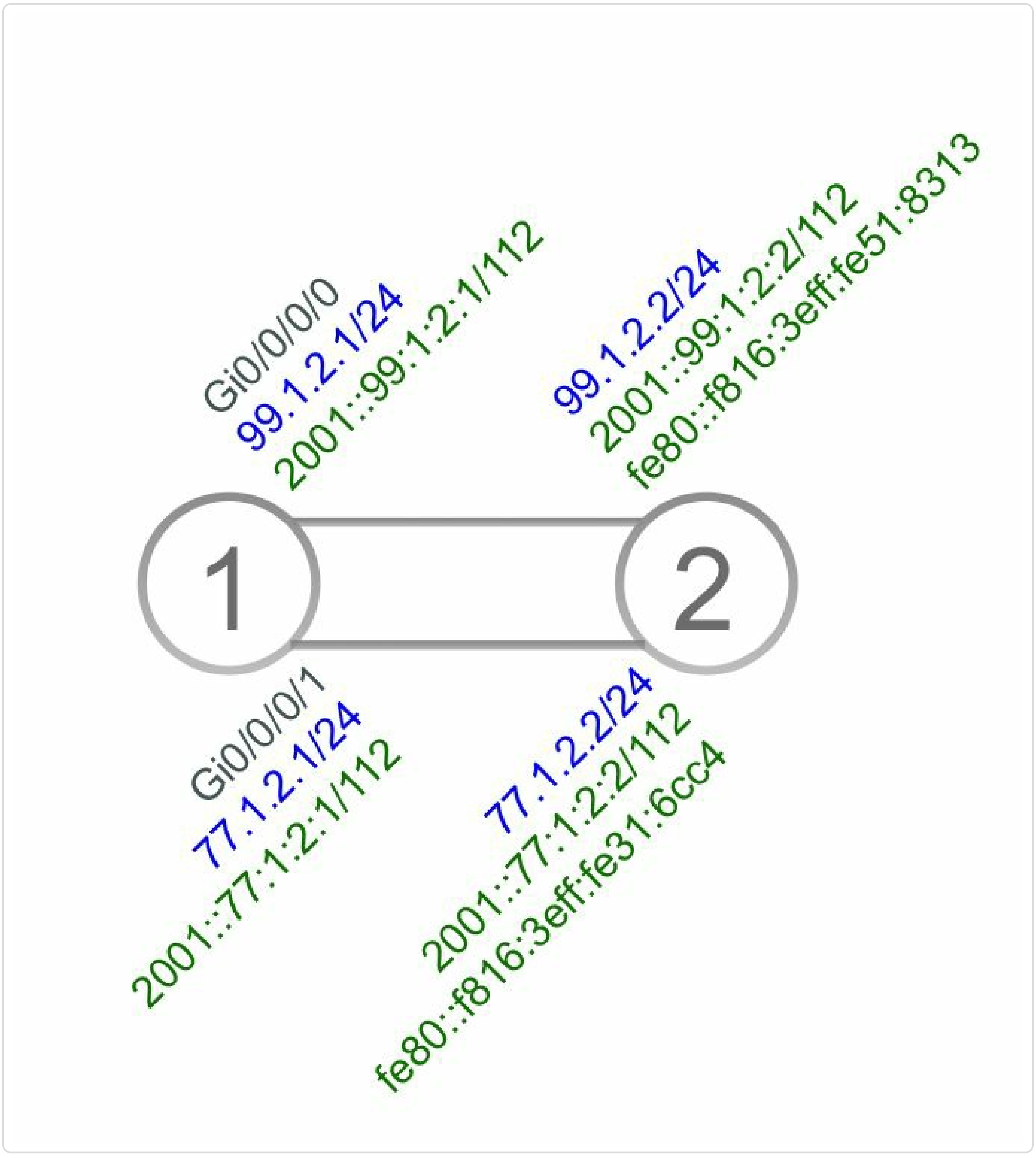


Figure 5-21: Adjacency-SID persistency – network topology

The sequence of events following an adjacency failure is illustrated for ISIS but the equivalent functionality exists for OSPF. For the purpose of this example, a second link is added between the two nodes. This parallel link provides a TI-LFA backup path for the protected Adjacency-SID. See the TI-LFA chapter 9 for more details of the TI-LFA functionality. For this example, it is enough to know that the Adjacency-SID is protected and a backup path is available. The failing adjacency is the adjacency on interface Gi0/0/0/0.

Example 5-33 shows the detailed information of the ISIS adjacency on interface Gi0/0/0/0. Two Adjacency-SIDs have been allocated per address family: one protection-eligible and one unprotected. 32102 and 33102 are the protected and unprotected IPv4 Adjacency-SIDs. 30102 and 31102 are the protected and unprotected IPv6 Adjacency-SIDs. Also note the details of the repair path of the protected Adjacency-SIDs.

Example 5-33: Adjacency-SID persistency – Adjacency-SIDs

```
RP/0/0/CPU0:xrvr-1#show isis adjacency detail Gi0/0/0/0 level 2

IS-IS 1 Level-2 adjacencies:
System Id      Interface        SNPA           State Hold
Changed  NSF IPv4 IPv6


 BFD  BFD
xrvr-2         Gi0/0/0/0        *PtoP*         Up    21
00:02:04 Yes None None
  Area Address:           49
  Neighbor IPv4 Address:  99.1.2.2*
  **Adjacency SID:          32102 (protected)**
   Backup label stack:    [ImpNull]
   Backup stack size:     1
   Backup interface:      Gi0/0/0/1
   Backup nexthop:        77.1.2.2
   Backup node address:   1.1.1.2
  **Non-FRR Adjacency SID:  33102**
  Neighbor IPv6 Address:  fe80::f816:3eff:fe51:8313*
  **Adjacency SID:          30102 (protected)**
   Backup label stack:    [ImpNull]
   Backup stack size:     1
   Backup interface:      Gi0/0/0/1
   Backup nexthop:        fe80::f816:3eff:fe31:6cc4
   Backup node address:   2001::1:1:1:2
  **Non-FRR Adjacency SID:  31102**
  Topology:               IPv4 Unicast
  Topology:               IPv6 Unicast
```

```
Total adjacency count: 1
```

In Example 5-34, one of the two links is failed by shutting down interface Gi0/0/0/0 on Node1.

Example 5-34: Adjacency-SID persistency – Failing link

```
RP/0/0/CPU0:xrvr-1#configure
RP/0/0/CPU0:xrvr-1(config)#interface Gi0/0/0/0
RP/0/0/CPU0:xrvr-1(config-if)#shutdown
RP/0/0/CPU0:xrvr-1(config-if)#commit
RP/0/0/CPU0:xrvr-1(config-if)#end
RP/0/0/CPU0:xrvr-1#
```

Immediately following the link failure, the labels of the unprotected Adjacency-SIDs (31102 and 33102) are deleted by IGP. See the output of `show mpls label table detail` in Example 5-35. The labels of the protected Adjacency-SIDs (30102 and 32102) are still `InUse`, steering the traffic over their backup path, while the labels 31102 and 33102 are no longer displayed.

Example 5-35: Adjacency-SID persistency – LSD table immediately after failure

```
RP/0/0/CPU0:xrvr-1#show mpls label table detail
Table Label   Owner                           State  Rewrite
----- ------- ------------------------------- ------ -------
<...>
0     30102   ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe51:8313)
0     32102   ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=99.1.2.2)
<...>
```

After the IGP delay period, IGP also deletes the protected Adjacency-SIDs' labels. By that time it is expected that the traffic flows have been rerouted to another path, not using the Adjacency-SID of the failed adjacency. The output of `show mpls label table` in Example 5-36 does not show any of the above Adjacency-SID labels.

Example 5-36: Adjacency-SID persistency – LSD table after failure

```
RP/0/0/CPU0:xrvr-1#show mpls label table label 30102
RP/0/0/CPU0:xrvr-1#
RP/0/0/CPU0:xrvr-1#show mpls label table label 32102
RP/0/0/CPU0:xrvr-1#
```

Now the interface is brought up again by rolling back the last committed configuration on Node1. See the configuration rollback in Example 5-37. This interface restoration is done within 30 minutes after the shutdown.

Example 5-37: Adjacency-SID persistency – Restoring link

```
RP/0/0/CPU0:xrvr-1#rollback configuration last 1

Loading Rollback Changes.
Loaded Rollback Changes in 1 sec
Committing.RP/0/0/CPU0:Feb  9 10:35:44.567 : ifmgr[224]:
%PKT_INFRA-LINK-3-UPDOWN : Interface GigabitEthernet0/0/0/0,
changed state to Down
RP/0/0/CPU0:Feb  9 10:35:44.607 : ifmgr[224]: %PKT_INFRA-LINK-
3-UPDOWN : Interface GigabitEthernet0/0/0/0, changed state to
Up

2 items committed in 1 sec (1)items/sec
Updating.
Updated Commit database in 1 sec
Configuration successfully rolled back 1 commits.
RP/0/0/CPU0:xrvr-1#
```

After restoring the link, ISIS brings up the adjacency over the link and ISIS asks LSD to allocate Adjacency-SID labels. The same labels are again allocated to the Adjacency-SIDs as can be verified in the LSD label table shown in Example 5-38.

Example 5-38: Adjacency-SID persistency – LSD table after restoration

```
RP/0/0/CPU0:xrvr-1#show mpls label table detail
Table Label   Owner                           State  Rewrite
----- ------- ------------------------------- ------ -------
<...>
0     30102   ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe51:8313)
0     31102   ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=3, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe51:8313)
0     32102   ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=99.1.2.2)
0     33102   ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=3, type=0,
intf=Gi0/0/0/0, nh=99.1.2.2)
<...>
```

HIGHLIGHT

IGPs automatically allocate Adjacency-SIDs (local labels from the dynamic MPLS label pool) for all interfaces when Segment Routing is enabled.

For each adjacency, two Adjacency-SIDs are allocated: one protection-eligible that can also have a backup path for handling adjacency failures, and another unprotected one.

Adjacency-SID persistency on Cisco IOS-XR implementations ensures that the same local label value is allocated to a given Adjacency Segment across transient event like link flaps.

Adjacency-SID protection on Cisco IOS-XR implementations using TI-LFA ensures that a backup path is available for any services using that Adjacency-SID label, thus allowing applications and controllers time to update their SR paths following a failure such that the services will not suffer from outage for an extended period of time.

"Adjacency-SIDs are to be used very sparingly and only in specific scenarios in SR like TI-LFA. The labels for Adjacency-SIDs may not be the same across a system reload, so it is better to never use these labels directly unless there are mechanisms in place that detect their changes and updates them (e.g. for TI-LFA). This is different from Prefix-SIDs which tend to be global and stable labels – hence used more widely in a number of use-cases."

— Ketan Talaulikar

5.4.5 ISIS Adjacency-SID

Figure 5-22 shows a two-node network topology that will be used in this section.

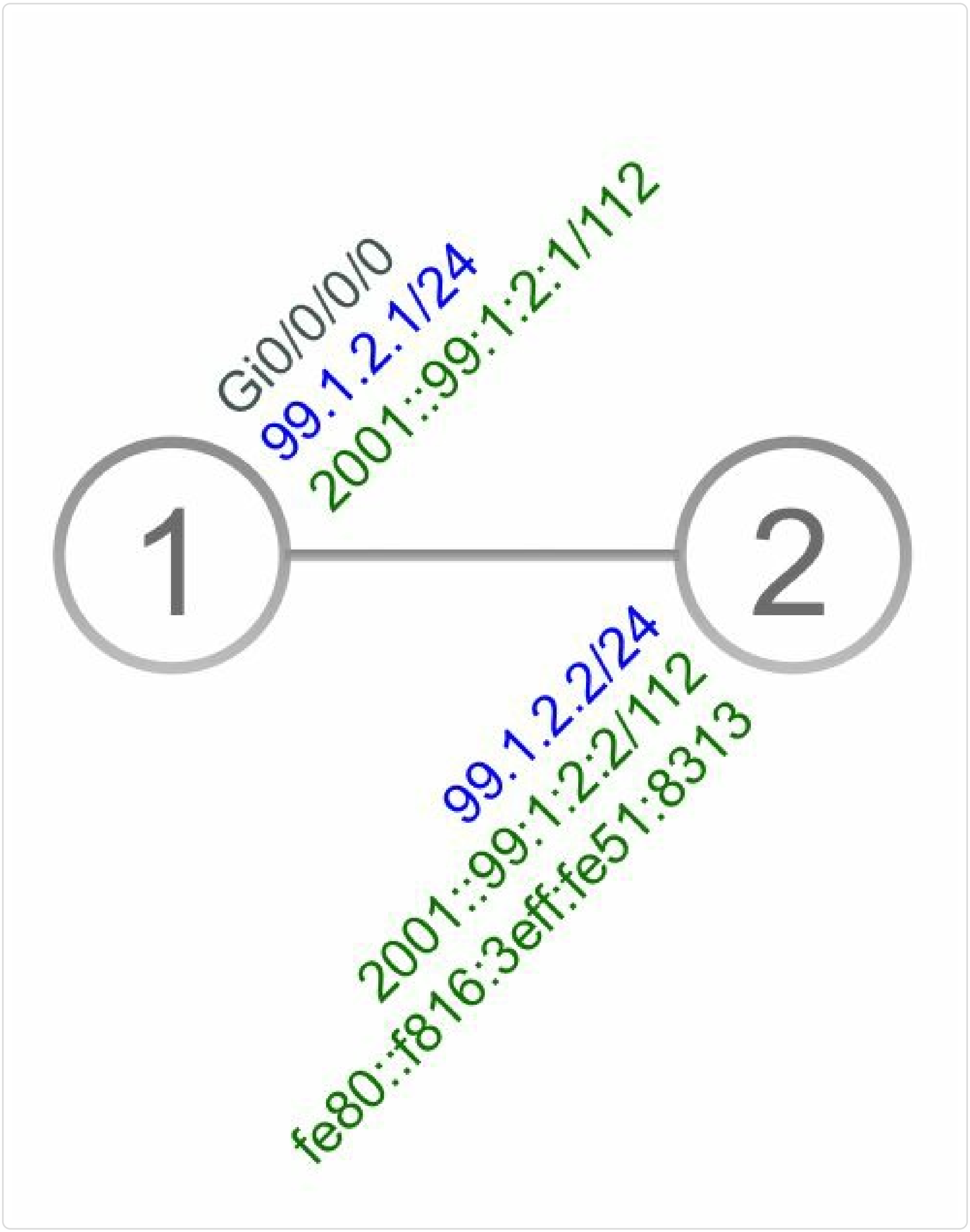


Figure 5-22: ISIS Adjacency-SID – network topology

Segment Routing is enabled for ISIS on both nodes in this topology and ISIS automatically allocates Adjacency-SIDs for all its adjacencies. The adjacency in this example is a point-to-point adjacency. Adjacencies on a

LAN network are covered further in this chapter. Both nodes are L1L2 nodes, so there is a Level-2 as well as a Level-1 adjacency for each address-family between the two nodes. The multi-topology model is used: IPv4 and IPv6 use a different, possibly non-congruent, topology. To simplify the output of the show commands, no prefixes are leaked between the levels. Therefor the route-policy DROP is used to prevent the default leaking from Level-1 into Level-2. In a real network this would break connectivity between the two levels. The configuration of Node1 is shown in Example 5-39.

Example 5-39: ISIS Adjacency-SID – router configuration Node1

```
route-policy DROP
  drop
end-policy
!
router isis 1
 is-type level-1-2
 net 49.0001.0000.0000.0001.00
 address-family ipv4 unicast
  metric-style wide
  !! prevent L1 to L2 leaking to simplify
  !! L2 is not propagated to L1 by default
  propagate level 1 into level 2 route-policy DROP
  segment-routing mpls
 !
 address-family ipv6 unicast
  metric-style wide
  !! prevent L1 to L2 leaking to simplify
  !! L2 is not propagated to L1 by default
  propagate level 1 into level 2 route-policy DROP
  segment-routing mpls
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
   prefix-sid absolute 16001
  !
  address-family ipv6 unicast
```

```
    prefix-sid absolute 17001
   !
  !
  interface GigabitEthernet0/0/0/0
   point-to-point
   address-family ipv4 unicast
   !
   address-family ipv6 unicast
   !
  !
 !
```

Example 5-40 shows the output of the command `show isis adjacency detail`. The output shows that two Adjacency-SIDs are allocated for each level and each topology, IPv4 and IPv6. Each adjacency gets a protection-eligible Adjacency-SID and an unprotected Adjacency-SID (named `Non-FRR Adjacency-SID` in the output). Note that this single-connected 2-node topology does not provide any backup possibility, so the Adjacency-SIDs are not actually protected in this case. If the Adjacency-SID would be protected, the keyword `(protected)` would be shown in the output, as well as the actual backup path.

Note that different Adjacency-SIDs are allocated for the Level-1 and Level-2 adjacencies.

The output in Example 5-40 would be the same in the case that IPv4/IPv6 single-topology mode would be used.

Example 5-40: ISIS Adjacency-SID – ISIS adjacency

```
RP/0/0/CPU0:xrvr-1#show isis adjacency detail systemid xrvr-2

IS-IS 1 Level-1 adjacencies:
System Id      Interface        SNPA           State Hold
Changed  NSF IPv4 IPv6
```

```
 BFD  BFD
xrvr-2         Gi0/0/0/0        *PtoP*         Up    25
00:00:28 Yes None None
  Area Address:           49.0001
  Neighbor IPv4 Address:  99.1.2.2*
  Adjacency SID:          34102
  Non-FRR Adjacency SID:  35102
  Neighbor IPv6 Address:  fe80::f816:3eff:fe51:8313*
  Adjacency SID:          36102
  Non-FRR Adjacency SID:  37102
  Topology:               IPv4 Unicast
  Topology:               IPv6 Unicast

Total adjacency count: 1

IS-IS 1 Level-2 adjacencies:
System Id      Interface        SNPA           State Hold
Changed  NSF IPv4 IPv6

 BFD  BFD
xrvr-2         Gi0/0/0/0        *PtoP*         Up    25
00:00:28 Yes None None
  Area Address:           49.0001
  Neighbor IPv4 Address:  99.1.2.2*
  Adjacency SID:          30102
  Non-FRR Adjacency SID:  31102
  Neighbor IPv6 Address:  fe80::f816:3eff:fe51:8313*
  Adjacency SID:          32102
  Non-FRR Adjacency SID:  33102
  Topology:               IPv4 Unicast
  Topology:               IPv6 Unicast

Total adjacency count: 1
```

Example 5-41 shows the MPLS forwarding entries of the Adjacency-SIDs, using the command `show mpls forwarding`. ISIS uses an index to distinguish the different Adjacency-SIDs for adjacencies between neighbors over the same interface. This is indicated as `(idx n)` in the third column of the output. This index is only used internally and must not be confused with the SID index of Prefix Segments. The protected

Adjacency-SIDs for Level-1 and Level-2 have index 0 and index 1 respectively. The unprotected Adjacency-SIDs for Level-1 and Level-2 have index 2 and index 3 respectively. The IPv4 and IPv6 Adjacency-SIDs can be distinguished in the output by the fifth column, `Next Hop`.

Example 5-41: ISIS Adjacency-SID – MPLS forwarding table

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 30000 40000
Local  Outgoing    Prefix             Outgoing     Next
Hop        Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
30102  Pop         SR Adj (idx 1)     Gi0/0/0/0
99.1.2.2        0
31102  Pop         SR Adj (idx 3)     Gi0/0/0/0
99.1.2.2        0
32102  Pop         SR Adj (idx 1)     Gi0/0/0/0
fe80::f816:3eff:fe51:8313   \

 0
33102  Pop         SR Adj (idx 3)     Gi0/0/0/0
fe80::f816:3eff:fe51:8313   \

 0
34102  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.1.2.2        0
35102  Pop         SR Adj (idx 2)     Gi0/0/0/0
99.1.2.2        0
36102  Pop         SR Adj (idx 0)     Gi0/0/0/0
fe80::f816:3eff:fe51:8313   \

                                     0
37102  Pop         SR Adj (idx 2)     Gi0/0/0/0
fe80::f816:3eff:fe51:8313   \

 0
```

Example 5-42 shows the LSD label entries using the command `show mpls label table detail`. The detailed output also shows the label context of each label..

Example 5-42: ISIS Adjacency-SID – MPLS label table

```
RP/0/0/CPU0:xrvr-1#show mpls label table detail
Table Label    Owner                           State  Rewrite
----- -------- ------------------------------- ------ -------
<...>
0     30102    ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=99.1.2.2)
0     31102    ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=3, type=0,
intf=Gi0/0/0/0, nh=99.1.2.2)
0     32102    ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe51:8313)
0     33102    ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=3, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe51:8313)
0     34102    ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=0,
intf=Gi0/0/0/0, nh=99.1.2.2)
0     35102    ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=2, type=0,
intf=Gi0/0/0/0, nh=99.1.2.2)
0     36102    ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=0, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe51:8313)
0     37102    ISIS(A):1                       InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=2, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe51:8313)
```

5.4.5.1 IS-IS Adjacency-SID Advertisement

ISIS advertises the Adjacency-SID labels by attaching an Adjacency-SID sub-TLV to the IS Reachability TLV of the adjacency it is associated with. ISIS adjacencies are advertised using one of the IS Reachability TLVs:

Extended IS Reachability TLV (type 22) and Multi-Topology IS Reachability TLV (type 222). Adjacency-SID sub-TLVs can also be added to IS Neighbor Attribute TLV (type 23) and MT IS Neighbor Attribute TLV (type 223), but this is currently not done in Cisco IOS XR..

The Adjacency-SID sub-TLV has the format shown in Figure 5-23.

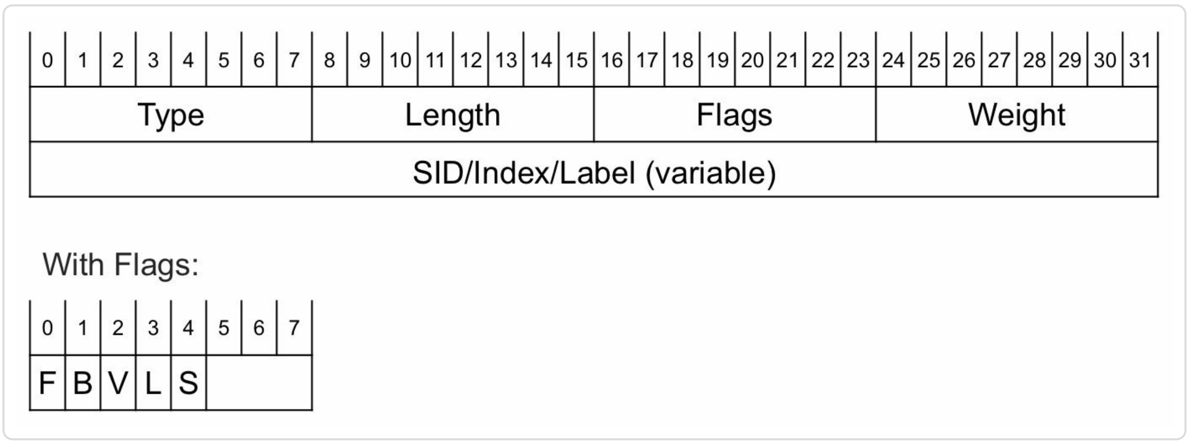


Figure 5-23: ISIS Adjacency-SID sub-TLV format

The Adjacency-SID Sub-TLV contains the following fields:

- Flags field:
  - F (Address-Family): if unset, then Adjacency-SID uses IPv4 encapsulation; if set, then Adjacency-SID uses IPv6 encapsulation
  - B (Backup): if set, then the Adjacency-SID is protection-eligible (e.g. Using TI-LFA); if unset, then the Adjacency-SID is unprotected
  - V (Value): set if Adjacency-SID carries an absolute value, unset if the Adjacency-SID carries an index – IOS XR: always set
  - L (Local/Global): set if Adjacency-SID has local significance, unset if Adjacency-SID has global significance – IOS XR: always set

    - S (Set): set if Adjacency-SID refers to a set of adjacencies – IOS XR: always unset
- Weight field: The value represents the weight of the Adjacency-SID for the purpose of load balancing – Weight = 0 in IOS XR
- SID/Index/Label: with V-flag=1 and L-flag=1 the 20 rightmost bits are used to encode a MPLS label value. This is the only supported combination of flags in IOS XR.

The network topology of Figure 5-22 is used to illustrate Adjacency-SID advertisement in ISIS.

Example 5-43 shows the output of the command `show isis database verbose level 1`, showing the ISIS LSP originated in Level-1 by Node1. Example 5-44 shows the Level-2 LSP originated by Node1. The IS-Extended TLV (type 22) for the IPv4 adjacency includes the Adjacency-SID sub-TLV, as well as the IPv4 interface address sub-TLV and IPv4 neighbor address sub-TLV. These last two sub-TLVs are defined in the ISIS TE Extensions specification (IETF RFC 3784) and are used to identify the link of the adjacency in case multiple parallel adjacencies exist between two nodes.

The Adjacency-SIDs for IPv4 have the F-flag (Address-family) unset. For the IPv6 Adjacency-SIDs this flag is set. The B-flag (Backup) indicates if the Adjacency-SID is protection-eligible (B:1) or not (B:0). All Adjacency-SIDs have the V-flag (Value) and L-flag (Local) set, which means that they are Local Segments and contain an absolute label value. The S-flag (Set) is not set, indicating they are not referring to a set of adjacencies.

Example 5-43: ISIS Adjacency-SID – Level-1 advertisement example

```
RP/0/0/CPU0:xrvr-1#show isis database level 1 xrvr-1 verbose

IS-IS 1 (Level-1) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x00000009   0x58c6
613            0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6
Unicast                                    0/0/0
  Hostname:     xrvr-1
  IP Address:   1.1.1.1
  IPv6 Address: 2001::1:1:1:1
  Router Cap:   1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 10         IS-Extended xrvr-2.00
    Interface IP Address: 99.1.2.1
    Neighbor IP Address: 99.1.2.2
    ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:34102
    ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:35102
  Metric: 0          IP-Extended 1.1.1.1/32
    Prefix-SID Index: 1, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 10         IP-Extended 99.1.2.0/24
  Metric: 10         MT (IPv6 Unicast) IS-Extended xrvr-2.00
    Interface IPv6 Address: 2001::99:1:2:1
    Neighbor IPv6 Address: 2001::99:1:2:2
    ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:36102
    ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:37102
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
    Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 10         MT (IPv6 Unicast) IPv6 2001::99:1:2:0/112

 Total Level-1 LSP count: 1     Local Level-1 LSP count: 1
```

Example 5-44: ISIS Adjacency-SID – Level-2 advertisement example

```
RP/0/0/CPU0:xrvr-1#show isis database level 2 xrvr-1 verbose

IS-IS 1 (Level-2) Link State Database
```

```
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x000009e2   0x92c5
823            0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6
Unicast                                      0/0/0
  Hostname:     xrvr-1
  IP Address:   1.1.1.1
  IPv6 Address: 2001::1:1:1:1
  Router Cap:   1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 10         IS-Extended xrvr-2.00
    Interface IP Address: 99.1.2.1
    Neighbor IP Address: 99.1.2.2
    ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:30102
    ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:31102
  Metric: 0          IP-Extended 1.1.1.1/32
    Prefix-SID Index: 1, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 10         IP-Extended 99.1.2.0/24
  Metric: 10         MT (IPv6 Unicast) IS-Extended xrvr-2.00
    Interface IPv6 Address: 2001::99:1:2:1
    Neighbor IPv6 Address: 2001::99:1:2:2
    ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:32102
    ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:33102
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
    Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 10         MT (IPv6 Unicast) IPv6 2001::99:1:2:0/112

 Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

When the single-topology model is used, with congruent IPv4 and IPv6 topologies, then a single adjacency is advertised for both IPv4 and IPv6. This means that both IPv4 and IPv6 Adjacency-SIDs must be attached to the IS reachability TLV advertising this single adjacency. The output in

Example 5-45 and Example 5-46 illustrate this for Level-1 and Level-2 respectively.

Example 5-45: ISIS Adjacency-SID – Level-1 single-topology advertisement example

```
RP/0/0/CPU0:xrvr-1#show isis database level 1 xrvr-1 verbose

IS-IS 1 (Level-1) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x0000000e   0x17ed
1195            0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  Hostname:     xrvr-1
  IP Address:   1.1.1.1
  IPv6 Address: 2001::1:1:1:1
  Router Cap:   1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 10          IS-Extended xrvr-2.00
    Interface IP Address: 99.1.2.1
    Neighbor IP Address: 99.1.2.2
    Interface IPv6 Address: 2001::99:1:2:1
    Neighbor IPv6 Address: 2001::99:1:2:2
    ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:34102
    ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:35102
    ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:36102
    ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:37102
  Metric: 0           IP-Extended 1.1.1.1/32
    Prefix-SID Index: 1, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 10          IP-Extended 99.1.2.0/24
  Metric: 0           IPv6 2001::1:1:1:1/128
    Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 10          IPv6 2001::99:1:2:0/112

 Total Level-1 LSP count: 1     Local Level-1 LSP count: 1
```

Example 5-46: ISIS Adjacency-SID – Level-2 single-topology advertisement example

```
RP/0/0/CPU0:xrvr-1#show isis database level 2 xrvr-1 verbose
```

```
IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x000009e7   0xff3e
1195            0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  Hostname:     xrvr-1
  IP Address:   1.1.1.1
  IPv6 Address: 2001::1:1:1:1
  Router Cap:   1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 10          IS-Extended xrvr-2.00
    Interface IP Address: 99.1.2.1
    Neighbor IP Address: 99.1.2.2
    Interface IPv6 Address: 2001::99:1:2:1
    Neighbor IPv6 Address: 2001::99:1:2:2
    ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:30102
    ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:31102
    ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:32102
    ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:33102
  Metric: 0           IP-Extended 1.1.1.1/32
    Prefix-SID Index: 1, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 10          IP-Extended 99.1.2.0/24
  Metric: 0           IPv6 2001::1:1:1:1/128
    Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 10          IPv6 2001::99:1:2:0/112

 Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

5.4.5.2 IS-IS LAN Adjacency-SID

The ISIS nodes on a LAN elect a Designated Intermediate System (DIS) to play the role of the pseudonode. The DIS election mechanism and how ISIS floods and maintains adjacencies on a LAN is out of scope for this book.

The network topology in Figure 5-24 consists of a LAN segment with three nodes connected to it. Node3 is elected as Designated Intermediate System (DIS).

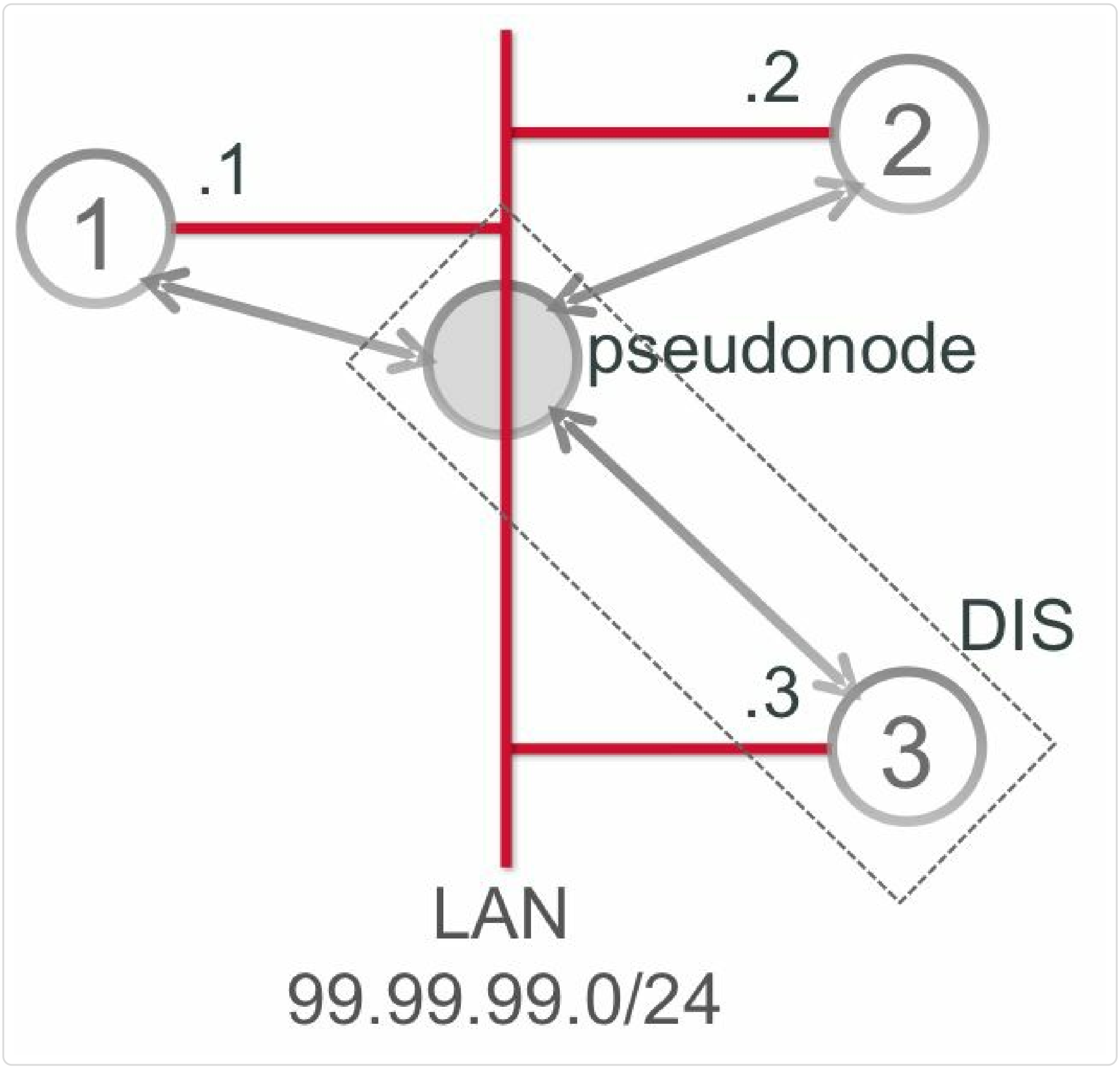


Figure 5-24: LAN with ISIS DIS

The DIS represents the pseudonode; it creates and updates the pseudonode LSP and floods LSPs on the LAN. Each ISIS node on the LAN maintains a full adjacency with the other nodes on the LAN. See the output of `show isis adjacency` on the non-DIS node Node1 in Example 5-47. Node1 has an adjacency to Node2 and Node3 on interface Gi0/0/0/0. These adjacencies are LAN adjacencies, they show the MAC address as Subnetwork Point of Attachment (SNPA) instead of `PtoP` in the SNPA column.

Example 5-47: ISIS adjacencies on LAN

```
RP/0/0/CPU0:xrvr-1#show isis adjacency

IS-IS 1 Level-2 adjacencies:
System Id      Interface        SNPA           State Hold
Changed  NSF IPv4 IPv6

 BFD  BFD
xrvr-2         Gi0/0/0/0        fa16.3e9a.29a8 Up    25
00:00:23 Yes None None
xrvr-3         Gi0/0/0/0        fa16.3eb9.643e Up    9
  00:00:29 Yes None None

Total adjacency count: 2
```

ISIS allocates two Adjacency-SIDs (one protection-eligible and one unprotected) for each adjacency to the other nodes on the LAN. A protection-eligible Adjacency-SID is allocated, but currently LAN Adjacency-SIDs are not TI-LFA protected. The Adjacency-SID labels are displayed in the `show isis adjacency detail` output collected on Node1. See Example 5-48.

Example 5-48: Detailed ISIS adjacencies on LAN

```
RP/0/0/CPU0:xrvr-1#show isis adjacency detail

IS-IS 1 Level-2 adjacencies:
System Id      Interface        SNPA           State Hold
Changed  NSF IPv4 IPv6

 BFD  BFD
xrvr-2         Gi0/0/0/0        fa16.3e9a.29a8 Up    29
00:01:29 Yes None None
  Area Address:           49.0001
```

```
    Neighbor IPv4 Address:   99.99.99.2*
    Adjacency SID:           30102
    Non-FRR Adjacency SID:   31102
    Neighbor IPv6 Address:   fe80::f816:3eff:fe9a:29a8*
    Adjacency SID:           32102
    Non-FRR Adjacency SID:   33102
    DIS Priority:            64
    Local Priority:          64
    Neighbor Priority:       64
    Topology:                IPv4 Unicast
    Topology:                IPv6 Unicast
  xrvr-3         Gi0/0/0/0        fa16.3eb9.643e Up    8
  00:01:35 Yes None None
    Area Address:            49.0001
    Neighbor IPv4 Address:   99.99.99.3*
    Adjacency SID:           30103
    Non-FRR Adjacency SID:   31103
    Neighbor IPv6 Address:   fe80::f816:3eff:feb9:643e*
    Adjacency SID:           32103
    Non-FRR Adjacency SID:   33103
    DIS Priority:            64
    Local Priority:          64
    Neighbor Priority:       64 (DIS)
    Topology:                IPv4 Unicast
    Topology:                IPv6 Unicast

  Total adjacency count: 2
```

For example, Node1 allocates IPv4 Adjacency-SIDs 30102 and 31102 for its IPv4 adjacency to Node2 and IPv6 Adjacency-SIDs 32102 and 33102 for its IPv6 adjacency to Node3.

The forwarding entries for these Adjacency-SIDs are programmed in the MPLS forwarding table, as illustrated in Example 5-49. The output shows the label entries on Node1 for the Adjacency-SIDs to the two other nodes on the LAN (Node2 and Node3). These are the labels that are shown in the output of Example 5-48. Notice that the Adjacency-SID labels steer traffic to the same outgoing interface (Gi0/0/0/0), but to different next-hops (e.g.

IPv4 99.99.99.2 to Node2 and 99.99.99.3 to Node3). These next-hops are the interface addresses of the remote nodes on the LAN.

Example 5-49: LAN Adjacency-SID MPLS forwarding entries

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 30000 40000
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                   Switched
------ ----------- ------------------ ------------ ------------
--- ------------
30102  Pop         SR Adj (idx 1)     Gi0/0/0/0
99.99.99.2      0
31102  Pop         SR Adj (idx 3)     Gi0/0/0/0
99.99.99.2      0
32102  Pop         SR Adj (idx 1)     Gi0/0/0/0
fe80::f816:3eff:fe9a:29a8   \

 0
33102  Pop         SR Adj (idx 3)     Gi0/0/0/0
fe80::f816:3eff:fe9a:29a8   \

 0
30103  Pop         SR Adj (idx 1)     Gi0/0/0/0
99.99.99.3      0
31103  Pop         SR Adj (idx 3)     Gi0/0/0/0
99.99.99.3      0
32103  Pop         SR Adj (idx 1)     Gi0/0/0/0
fe80::f816:3eff:feb9:643e   \

 0
33103  Pop         SR Adj (idx 3)     Gi0/0/0/0
fe80::f816:3eff:feb9:643e   \

 0
```

Example 5-50 shows the LSD label entries for the Adjacency-SIDs, with their label context.

Example 5-50: ISIS LAN Adjacency-SIDs – MPLS label table

```
RP/0/0/CPU0:xrvr-1#show mpls label table detail
Table Label    Owner                          State  Rewrite
----- -------- ------------------------------ ------ -------
0     0        LSD(A)                         InUse  Yes
0     1        LSD(A)                         InUse  Yes
0     2        LSD(A)                         InUse  Yes
0     13       LSD(A)                         InUse  Yes
0     16000    ISIS(A):1                      InUse  No
  (Lbl-blk SRGB, vers:0, (start_label=16000, size=8000)
0     30102    ISIS(A):1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=99.99.99.2)
0     31102    ISIS(A):1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=3, type=0,
intf=Gi0/0/0/0, nh=99.99.99.2)
0     32102    ISIS(A):1                      InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe9a:29a8)
0     33102    ISIS(A):1                      InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=3, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:fe9a:29a8)
0     30103    ISIS(A):1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=99.99.99.3)
0     31103    ISIS(A):1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=3, type=0,
intf=Gi0/0/0/0, nh=99.99.99.3)
0     32103    ISIS(A):1                      InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=1, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:feb9:643e)
0     33103    ISIS(A):1                      InUse  Yes
  (SR Adj Segment IPv6, vers:0, index=3, type=0,
intf=Gi0/0/0/0, nh=fe80::f816:3eff:feb9:643e)
```

5.4.5.3 ISIS LAN Adjacency-SID Advertisement

The DIS generates the LSP for the pseudonode, advertising the pseudonode’s adjacencies to all the nodes on the LAN: Node1, Node2, and Node3. Example 5-51 illustrates the pseudonode LSP as generated by the DIS (Node3) on the LAN.

A pseudonode LSP can be recognized by its LSP-ID. For an LSP advertised by a DIS on behalf of a pseudonode, the LSP-ID has a non-zero Pseudonode-ID. The Pseudonode-ID is the single byte field after the dot in the LSP-ID. For example in the output of Example 5-51 the Pseudonode-ID is the “01” in the LSP-ID “xrvr-3.01-00”. The “01” indicates this LSP is generated on behalf of the pseudonode.

Example 5-51: ISIS DIS (pseudonode) advertisement

```
RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-3.01-00

IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-3.01-00          0x00000001   0x4d41
935            0/0/0
  Metric: 0           IS-Extended xrvr-3.00
  Metric: 0           IS-Extended xrvr-1.00
  Metric: 0           IS-Extended xrvr-2.00
```

REMINDER: LSP-ID

The LSP-ID uniquely identifies an IS-IS LSP. The LSP-ID is a concatenation of the System-ID (6 bytes), the Pseudonode-ID (8 bits), and the Fragment-ID (8 bits). The System-ID is the unique identifier of the system, comparable to the OSPF router-ID. The Pseudonode-ID identifies if the LSP is generated for the real instance of the node or on behalf of a pseudonode. A Pseudonode-ID value 0 indicates the LSP is for the real instance, a non-zero value means the LSP is for a pseudonode. The Fragment-ID is used to identify fragments of an LSP. The LSP-ID is written in the show output as “System ID or name”.”Pseudonode ID”-“Fragment ID”. The Node-ID is the concatenation of the System-ID and the Pseudonode-ID.

Each node on the LAN advertises its adjacency to the pseudonode. Example 5-52 shows the LSP generated by Node1. Node1 advertises its adjacency to the pseudonode in both topologies, IPv4 and IPv6, these are highlighted in the output. For simplicity, Example 5-52 shows the `detail`'ed ISIS database output, not the `verbose` output. Therefore the Segment Routing information is not displayed in this output. The `verbose` output is shown in Example 5-53 after the LAN Adjacency-SID advertisement format is explained.

Example 5-52: ISIS adjacency to pseudonode advertisement

```
RP/0/0/CPU0:xrvr-1#show isis database xrvr-1 detail

IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x00000071   0xe250
1146            0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6
Unicast                                 0/0/0
  Hostname:     xrvr-1
  IP Address:   1.1.1.1
  IPv6 Address: 2001::1:1:1:1
  Router Cap:   1.1.1.1, D:0, S:0
  Metric: 10         IS-Extended xrvr-3.01
  Metric: 0          IP-Extended 1.1.1.1/32
  Metric: 10         IP-Extended 99.99.99.0/24
  Metric: 10         MT (IPv6 Unicast) IS-Extended xrvr-3.01
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
  Metric: 10         MT (IPv6 Unicast) IPv6
2001::99:99:99:0/112

 Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

Each node allocates a LAN Adjacency-SID towards each of the other nodes on the LAN. The sub-TLVs of these LAN Adjacency-SIDs are attached to the IS Reachability TLV of the node's adjacency to the pseudonode.

The format of the LAN Adjacency-SID sub-TLV is shown in Figure 5-25

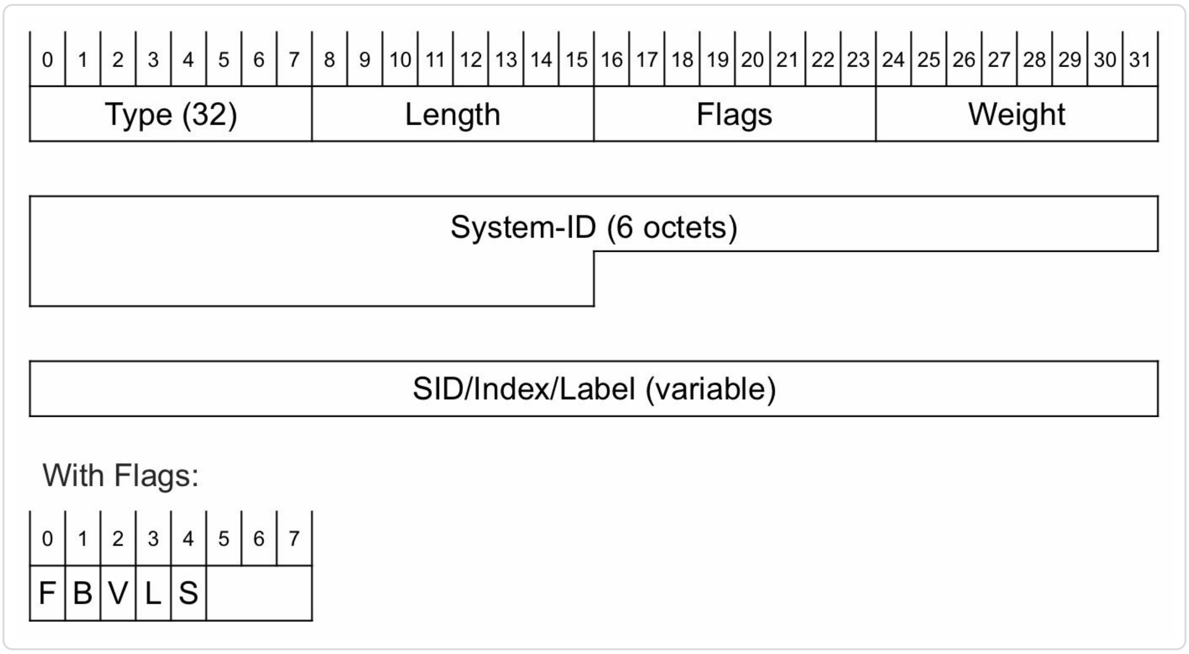


Figure 5-25: ISIS LAN Adjacency-SID sub-TLV format

The Adjacency-SID sub-TLV contains the following fields:

- The flags and Weight fields are identical to the fields in the Adjacency-SID sub-TLV. Please refer to section 5.4.5.1 "IS-IS Adjacency-SID Advertisement" for these fields.
- System-ID: ISIS System ID of the neighbor node
- SID/Label/Index: with V-flag=1 and L-flag=1 the 20 rightmost bits are used to encode a MPLS label value – always a MPLS label value (V:1, L:1) in Cisco IOS XR

Example 5-53 shows the LSP generated by Node1, including the Segment Routing information. For each other node on the LAN, Node1 adds a LAN Adjacency-SID sub-TLV in the IS Reachability TLV for its adjacency to the pseudonode. This is the case for both topologies, IPv4 and IPv6. In this example, Node3 is the DIS and the Node-ID of the pseudonode is "xrvr-

3.01”. The output shows that in the IPv6 topology four LAN Adjacency-SIDs are attached to the adjacency with the pseudonode: two Adjacency-SIDs towards Node2 (System-ID xrvr-2):

- one protection-eligible, line 25 (label 32102 with B-flag set)
- one unprotected, line 26 (label 33102 with B-flag unset)

And two Adjacency-SIDs towards Node3 (System-ID xrvr-3):

- one protection-eligible, line 27 (label 32103 with B-flag set)
- one unprotected, line 28 (label 33103 with B-flag unset)

Example 5-53: ISIS adjacency to pseudonode with Adjacency-SIDs

```
 1 RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-1
 2
 3 IS-IS 1 (Level-2) Link State Database
 4 LSPID                 LSP Seq Num  LSP Checksum  LSP
Holdtime  ATT/P/OL
 5 xrvr-1.00-00        * 0x00000071   0xe250
910            0/0/0
 6   Area Address: 49.0001
 7   NLPID:        0xcc
 8   NLPID:        0x8e
 9   MT:           Standard (IPv4 Unicast)
10   MT:           IPv6
Unicast                                    0/0/0
11   Hostname:     xrvr-1
12   IP Address:   1.1.1.1
13   IPv6 Address: 2001::1:1:1:1
14   Router Cap:   1.1.1.1, D:0, S:0
15     Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
16   Metric: 10         IS-Extended xrvr-3.01
17     LAN-ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:
30102 System ID:xrvr-2
18     LAN-ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:
31102 System ID:xrvr-2
19     LAN-ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:
30103 System ID:xrvr-3
```

```
     LAN-ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid: 31103 System ID:xrvr-3
   Metric: 0          IP-Extended 1.1.1.1/32
     Prefix-SID Index: 1, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
   Metric: 10         IP-Extended 99.99.99.0/24
   Metric: 10         MT (IPv6 Unicast) IS-Extended xrvr-3.01
     LAN-ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid: 32102 System ID:xrvr-2
     LAN-ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid: 33102 System ID:xrvr-2
     LAN-ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid: 32103 System ID:xrvr-3
     LAN-ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid: 33103 System ID:xrvr-3
   Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
     Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
   Metric: 10         MT (IPv6 Unicast) IPv6 2001::99:1:99:0/112

 Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

5.4.6 OSPFv2 Adjacency-SID

The two-node topology in Figure 5-26 is used to illustrate the Adjacency-SID advertisement in OSPF.

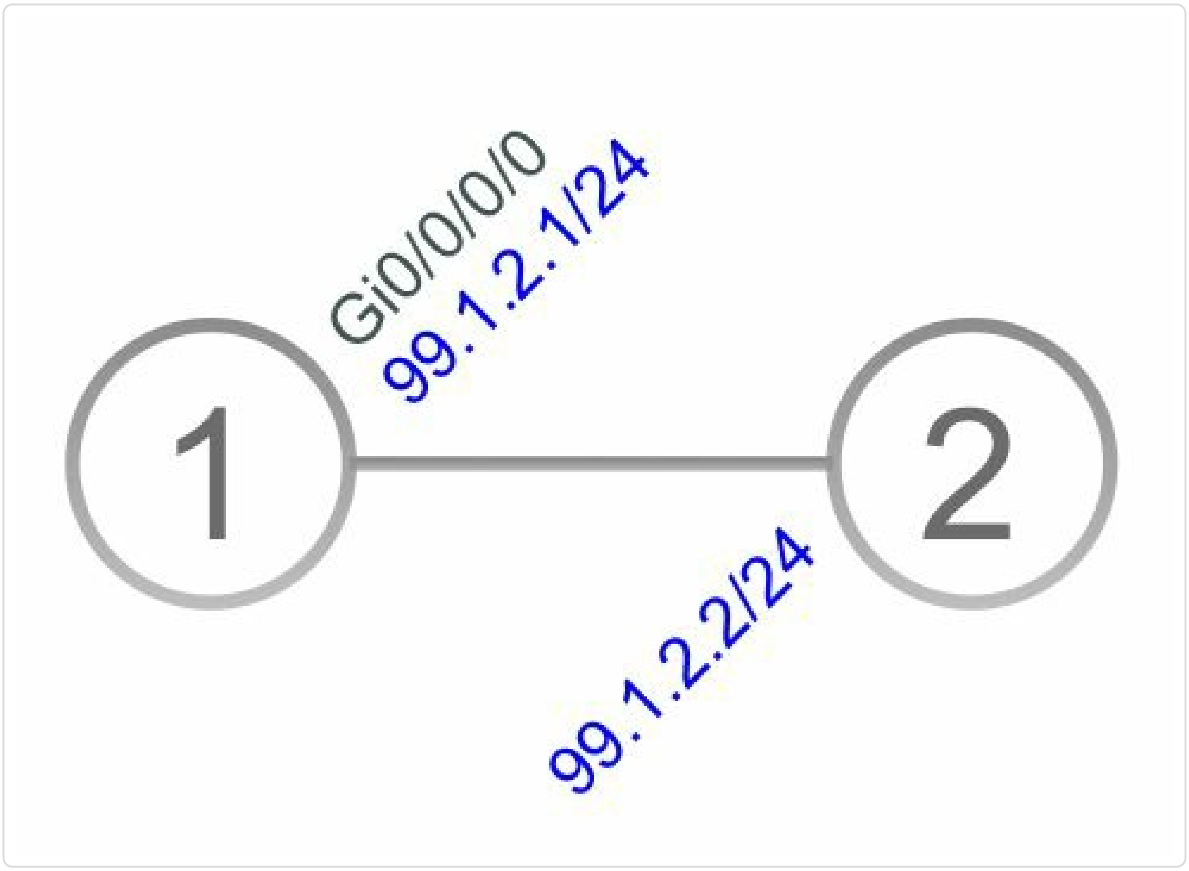


Figure 5-26: OSPFv2 Adjacency-SID – network topology

Segment Routing is enabled for OSPF on both nodes and OSPF automatically allocates Adjacency-SIDs for all its adjacencies. The adjacency in this example is a point-to-point adjacency. Adjacencies on a LAN network are covered later in this section.

The OSPF configuration of Node1 is displayed in Example 5-54.

Example 5-54: OSPFv2 Adjacency-SID – router configuration

```
router ospf 1
 router-id 1.1.1.1
 segment-routing mpls
 !!  segment-routing forwarding mpls !! by default
 area 0
  interface Loopback0
   passive enable
   prefix-sid absolute 16001
```

```
    !
    interface GigabitEthernet0/0/0/0
     cost 10
     network point-to-point
    !
    interface GigabitEthernet0/0/0/1
     cost 10
     network point-to-point
    !
   !
  !
```

Example 5-55 shows the output of the command `show ospf neighbor detail` on Node1. The output shows that two Adjacency-SIDs are allocated for the adjacency to Node2. Each adjacency gets a protection-eligible Adjacency-SID (`Adjacency SID Label`) and an unprotected Adjacency-SID (`Unprotected Adjacency SID Label` in the output). Note that this 2-node topology does not provide any backup path, so the Adjacency-SIDs are not actually protected in this case. If a protected Adjacency-SID is actually protected, then `Protected` is added to the output, e.g. `Adjacency SID Label: 30102, Protected.`

Example 5-55: OSPFv2 Adjacency-SID – Adjacency-SIDs

```
RP/0/0/CPU0:xrvr-1#show ospf neighbor 1.1.1.2 detail

* Indicates MADJ interface
# Indicates Neighbor awaiting BFD session up

Neighbors for OSPF 1

 Neighbor 1.1.1.2, interface address 99.1.2.2
    In the area 0 via interface GigabitEthernet0/0/0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:00:31
    Neighbor is up for 00:01:14
    Number of DBD retrans during last exchange 0
    Index 1/1, retransmission queue length 0, number of
retransmission 1
    First 0(0)/0(0) Next 0(0)/0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
    LS Ack list: NSR-sync pending 0, high water mark 0
    Adjacency SID Label: 30102
    Unprotected Adjacency SID Label: 31102

Total neighbor count: 1
```

Example 5-56 shows the MPLS forwarding entries of the Adjacency-SIDs, using the command `show mpls forwarding`. The protection-eligible and unprotected Adjacency-SIDs cannot be distinguished from this output, they both use `(idx 0)` in the output.

Example 5-56: OSPFv2 Adjacency-SID – MPLS forwarding

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 30102 31102
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
30102  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.1.2.2        0
31102  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.1.2.2        0
```

Example 5-57 shows the LSD label entries using the command `show mpls label table detail`. The detailed output also shows the label context of each label. The protection-eligible and unprotected Adjacency-SIDs are distinguished using the "type" key. Type=1 is the protection-eligible Adjacency-SID, type=2 the unprotected one.

Example 5-57: OSPFv2 Adjacency-SID – MPLS label table

```
RP/0/0/CPU0:xrvr-1#show mpls label table detail
Table Label   Owner                           State  Rewrite
----- ------- ------------------------------- ------ -------
<...>
0     30102   OSPF(A):ospf-1                  InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=1,
intf=Gi0/0/0/0, nh=99.1.2.2)
0     31102   OSPF(A):ospf-1                  InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=2,
intf=Gi0/0/0/0, nh=99.1.2.2)
```

5.4.6.1 OSPFv2 Adjacency-SID Advertisement

The Adjacency-SID information is advertised in OSPFv2 as a sub-TLV of the OSPFv2 Extended Link TLV as defined in IETF RFC 7684. This TLV is carried in an OSPFv2 Extended Link Opaque LSA (Opaque-Type 8) with area flooding scope (Type 10 LSA) to ensure that they are only flooded within the area on the same lines as the OSPF Router and Network LSAs which describe the links for the base OSPF protocol. The use of OSPFv2 Extended Link LSA is not limited to Segment Routing and this LSA can be used to advertise other additional link attributes.

The fields in the OSPFv2 Extended Link TLV describe the adjacency that the TLV applies to. This TLV has the format shown in Figure 5-27.

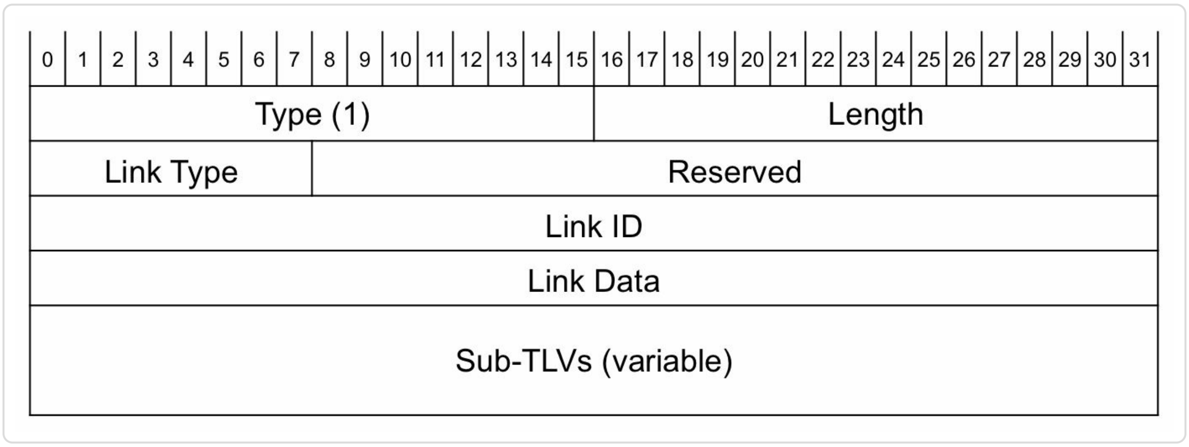


Figure 5-27: OSPFv2 Extended Link TLV format

The TLV contains the following fields:

- Link Type field: its value is as defined in the OSPFv2 specification (IETF RFC 2328, A.4.2):

Type	**Description**
1	Point-to-point connection to another router
2	Connection to a transit network
3	Connection to a stub network
4	Virtual link

- Link ID and Link Data fields: the meaning of the Link ID and Link Data fields depends on the Link Type field value (all according to the OSPFv2 specification IETF RFC 2328, A.4.2) as shown in Table 5-2. IOS XR does currently not use Link Type 4.

Table 5-2: OSPFv2 Extended Link TLV field values

Description	**Link Type**	**Link ID**	**Link Data**
Point-to-point connection (IP address assigned)	1	Neighbor’s Router ID	Interface IP address
Point-to-point connection (using IP unnumbered)			Interface ifIndex value
Connection to a transit network	2	IP address of Designated Router	Interface IP address
Connection to a stub network	3	Network IP address	

			Network IP subnet mask
Virtual link	4	Neighbor’s Router ID	Interface IP address

The Adjacency-SID sub-TLV is a sub-TLV of the Extended Link TLV and has the format shown in Figure 5-28.

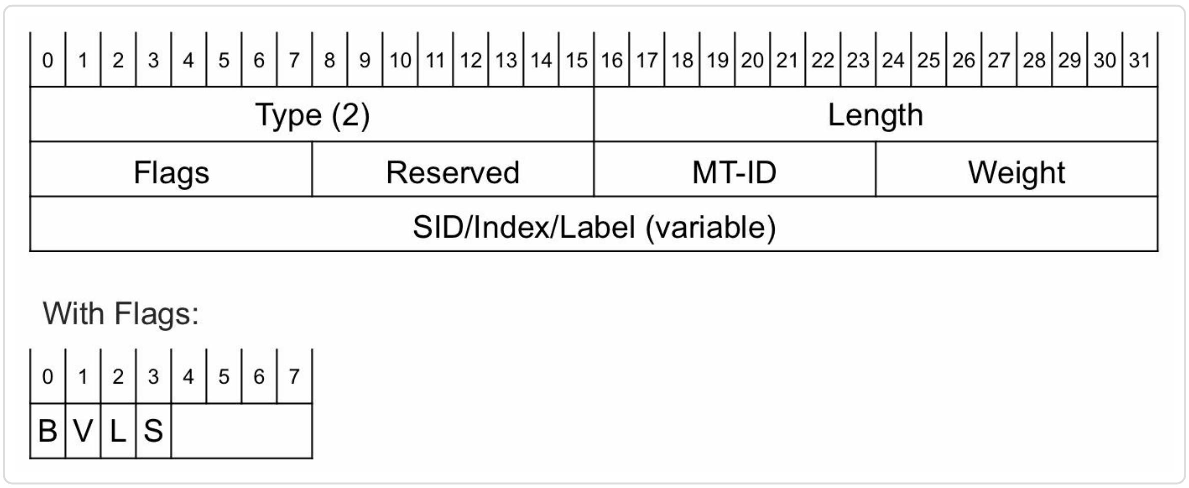


Figure 5-28: OSPFv2 Adjacency-SID sub-TLV format

The OSPFv2 Adjacency-SID sub-TLV contains the following fields:

- Flags field:
  - B (Backup): if set, then the Adjacency-SID is protection-eligible (e.g. Using TI-LFA); if unset, then the Adjacency-SID is unprotected
  - V (Value): set if Adjacency-SID carries an absolute value, unset if the Adjacency-SID carries an index – Cisco IOS XR: always set
  - L (Local/Global): set if Adjacency-SID has local significance, unset if Adjacency-SID has global significance – Cisco IOS XR: always set
  - S (Set): set if Adjacency-SID refers to a set of adjacencies – Cisco

IOS XR: always unset at the time of writing this book

- MT-ID field: Multi-Topology ID (as defined in IETF RFC 4915) – Cisco IOS XR: not supported at the time of writing this book
- Weight field: The value represents the weight of the Adjacency-SID for the purpose of load balancing – Weight = 0 in Cisco IOS XR
- SID/Index/Label field: with V-flag=1 and L-flag=1: the 20 rightmost bits are used to encode a MPLS label value – always a local MPLS label value (V:1 and L:1) in Cisco IOS XR

Example 5-58 shows the Router LSA (type 1 LSA) that is advertised by Node1. The topology is still the one in Figure 5-26. The Router LSA contains the point-to-point adjacency to Node2. The following fields identify this adjacency:

- Link Type = Point-to-point connection (1); “another Router (point-to-point)”
- Link ID = neighbor’s router-id (1.1.1.2)
- Link Data = local interface’s IP address 99.1.2.1 since it is not an unnumbered interface.

Example 5-58: OSPFv2 Router LSA example

```
RP/0/0/CPU0:xrvr-1#show ospf database router self-originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 0)

  LS age: 634
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.1.1.1
```

```
    Advertising Router: 1.1.1.1
    LS Seq Number: 80000008
    Checksum: 0x8891
    Length: 60
     Number of Links: 3

      Link connected to: a Stub Network
       (Link ID) Network/subnet number: 1.1.1.1
       (Link Data) Network Mask: 255.255.255.255
        Number of TOS metrics: 0
         TOS 0 Metrics: 1

      Link connected to: another Router (point-to-point)
       (Link ID) Neighboring Router ID: 1.1.1.2
       (Link Data) Router Interface address: 99.1.2.1
        Number of TOS metrics: 0
         TOS 0 Metrics: 10

      Link connected to: a Stub Network
       (Link ID) Network/subnet number: 99.1.2.0
       (Link Data) Network Mask: 255.255.255.0
        Number of TOS metrics: 0
         TOS 0 Metrics: 10
```

In this example, Node1 advertises only one Extended Link (Opaque-type 8) LSA since there is only one adjacency. This LSA happens to have Link State ID 8.0.0.3, where 8 is the Opaque Type and 0.0.3 = 3 the Opaque ID. The Opaque ID is a random identifier that is only used to distinguish between multiple Opaque Type 8 LSAs originated by the same node. See the output of Example 5-59.

Example 5-59: OSPFv2 Extended Link Opaque LSA example

```
RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 8.0.0.3


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 
0)
```

```

LS age: 1001
Options: (No TOS-capability, DC)
LS Type: Opaque Area Link
Link State ID: 8.0.0.3
Opaque Type: 8
Opaque ID: 3
Advertising Router: 1.1.1.1
LS Seq Number: 80000006
Checksum: 0x398a
Length: 68

  Extended Link TLV: Length: 44
    Link-type : 1             !! point-to-point connection
    Link ID   : 1.1.1.2       !! neighbor’s router-id
    Link Data : 99.1.2.1      !! interface’s IP address

   Adj sub-TLV: Length: 7
      Flags     : 0xe0        !! B:1, V:1, L:1, S:0
      MTID      : 0
      Weight    : 0
      Label     : 30102

   Adj sub-TLV: Length: 7
      Flags     : 0x60        !! B:0, V:1, L:1, S:0
      MTID      : 0
      Weight    : 0
      Label     : 31102

   Remote If Address sub-TLV: Length: 4
      Neighbor Address: 99.1.2.2
```

The Extended Link LSA contains the Extended Link TLV. The fields in this TLV specify the adjacency that it applies to. It uses the same fields as the ones used in the Router LSA for the adjacency:

- Link Type = Point-to-point connection (1)
- Link ID = neighbor’s router-id (1.1.1.2)
- Link Data = interface IP address 99.1.2.1 since the interface has an IP

address configured.

The Extended Link TLV contains two Adjacency-SID sub-TLVs, one for the protection-eligible Adjacency-SID (with B-flag:1) and one for the unprotected Adjacency-SID (with B-flag:0). Note that no protection is possible on this single connected 2-node network topology since there is no backup path available. The flags are displayed as a hexadecimal number in the OSPF database output. Both Adjacency-SIDs have the V-flag (Value) and L-flag (Local) set, which means that these are Local Segments and contain a label value. The S-flag (Set) is not set for either of the two, which means that they are not referring to a set of adjacencies.

The Extended Link TLV also includes the neighbor IP address, specified in a Remote Interface Address sub-TLV. This sub-TLV is only included for point-to-point adjacencies..

The TE Opaque LSAs were defined in RFC 3630, RFC 4203, and others to enable propagation of TE specific link attributes via OSPF. The information in these LSAs was typically meant to be originated and consumed by the TE control plane (namely RSVP-TE, GMPLS, etc.) – they are handled in an opaque manner by the OSPF protocol itself. Another implication of originating these LSAs is that it announces the link to be part of the classical MPLS-TE topology (e.g. for use by RSVP-TE) to all other nodes in the area. When OSPF was extended for Segment Routing, it was determined to be incorrect to use the same TE Opaque LSAs and instead, generic mechanisms using the OSPF Extended Prefix and Link Attribute LSAs were specified via RFC 7684. These new/generic LSA containers are for use by OSPF without any connotations with TE. The remote-address TLV that we saw along with the Adjacency-SID in the Extended Link Attribute LSA in Example 5-59 (lines 36 and 37), is an example of an existing TLV (defined for TE Opaque LSA) being re-used here. Other TLVs which could be re-used similarly would be related to SRLG, local/remote link identifiers, and extended metrics. This is done such that they may be used for OSPF protocol usage (e.g. for SR TE and TI-LFA) without interfering with the traditional RSVP-TE deployments. One can check the IETF draft-ppsenak-ospf-te-link-attr-reuse for more details and we would cover this in the next part of this book that deals with SR TE.

5.4.6.2 OSPF Multi-AreaAdjacency-SID

In the network topology of Figure 5-26, an OSPF area 1 is added on both nodes. There is still only one link between the two nodes. Interface Gi0/0/0/0 is added as a multi-area interface under area 1 on both nodes. See the configuration of Node1 in Example 5-60.

Example 5-60: Multi-area adjacency – configuration

```
router ospf 1
 router-id 1.1.1.1
 segment-routing mpls
 area 0
  interface Loopback0
   passive enable
   prefix-sid absolute 16001
  !
  interface GigabitEthernet0/0/0/0
   cost 10
   network point-to-point
  !
 !
 area 1
  multi-area-interface GigabitEthernet0/0/0/0
   cost 10
  !
 !
!
```

Both adjacencies are displayed in the output of the command `show ospf neighbor area-sorted` in Example 5-61. The Multi-Adjacency (MADJ) interface is marked with "*" at the end of the line.

Example 5-61: Multi-area adjacency – adjacencies

```
RP/0/0/CPU0:xrvr-1#show ospf neighbor area-sorted

* Indicates MADJ interface
# Indicates Neighbor awaiting BFD session up

Neighbors for OSPF 1

Area 0
Neighbor ID     Pri State    Dead Time Address         Up Time
 Interface
1.1.1.2         1   FULL/  - 00:00:37  99.1.2.2        00:56:13
Gi0/0/0/0

Total neighbor count: 1

Area 1
Neighbor ID     Pri State    Dead Time Address         Up Time
Interface
1.1.1.2         1   FULL/  - 00:00:34  99.1.2.2        00:00:53
Gi0/0/0/0*

Total neighbor count: 1
```

The output of `show ospf neighbor detailed` shows that the same Adjacency-SIDs are used for the adjacency in both areas.

Example 5-62: Multi-area adjacency – Adjacency-SIDs

```
RP/0/0/CPU0:xrvr-1#show ospf neighbor detail

* Indicates MADJ interface
# Indicates Neighbor awaiting BFD session up

Neighbors for OSPF 1
```

```
 Neighbor 1.1.1.2, interface address 99.1.2.2
    In the area 0 via interface GigabitEthernet0/0/0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:00:31
    Neighbor is up for 00:58:49
    Number of DBD retrans during last exchange 0
    Index 1/1, retransmission queue length 0, number of
retransmission 2
    First 0(0)/0(0) Next 0(0)/0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
    LS Ack list: NSR-sync pending 0, high water mark 0
    Adjacency SID Label: 30102
    Unprotected Adjacency SID Label: 31102

 Neighbor 1.1.1.2, interface address 99.1.2.2
    In the area 1 via interface GigabitEthernet0/0/0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:00:31
    Neighbor is up for 00:03:30
    Number of DBD retrans during last exchange 0
    Index 1/2, retransmission queue length 0, number of
retransmission 1
    First 0(0)/0(0) Next 0(0)/0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
    LS Ack list: NSR-sync pending 0, high water mark 0
    Adjacency SID Label: 30102
    Unprotected Adjacency SID Label: 31102


Total neighbor count: 2
```

The OSPF implementation in Cisco IOS-XR uses the same MPLS label for multi-area adjacencies to a given neighbor on the same interface so as to abstract this concept from the MPLS forwarding plane and reduce the

Adjacency-SID labels used. When it comes to forwarding the Segment Routing traffic over a link, the notion of area does not really apply since it is only a control plane concept. When it comes to programming the backup path for the protection-eligible Adjacency-SID for a multi-area adjacency, OSPF could choose any of the areas over which a backup path to the neighboring router is found (generally it would be in the backbone area).

The advertisement of the Multi-Area Adjacency in the non-primary area is almost identical to that of a regular, single-area adjacency. Example 5-63 shows the Router LSA as advertised by Node1 in area 1. This LSA only contains the adjacency and the link is considered unnumbered in that area. In the case of an unnumbered interface, as is the case here, the *ifIndex* of the interfaces is used in the Link Data field of the adjacency in the Router LSA. Example 5-64 shows how you can find the *ifIndex* of an interface. The *ifIndex* is shown in dotted decimal notation in the Router LSA's Link Data field.

Example 5-63: Multi-area adjacency – Router LSA

```
RP/0/0/CPU0:xrvr-1#show ospf 1 1 database router self-originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 1)

  LS age: 64
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.1.1.1
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000002
  Checksum: 0xac68
  Length: 36
  Area Border Router
  AS Boundary Router
```

```
    Number of Links: 1

     Link connected to: another Router (point-to-point)
      (Link ID) Neighboring Router ID: 1.1.1.2
      (Link Data) Router Interface address: 0.0.0.3
       Number of TOS metrics: 0
        TOS 0 Metrics: 10
```

Example 5-64: Multi-area adjacency –ifIndex

```
RP/0/0/CPU0:xrvr-1#show snmp interface gi0/0/0/0 ifindex
ifName : GigabitEthernet0/0/0/0  ifIndex : 3
```

The Extended Link TLV that contains the Adjacency-SID of the multi-area-adjacency in area 1 uses the following fields to identify the adjacency:

- Link Type = Point-to-point connection (1)
- Link ID = neighbor's router-id (1.1.1.2)
- Link Data = *ifIndex* of local interface since it is an unnumbered interface

See the output in Example 5-65. Apart from the different Link Data field in the Extended Link TLV, the content is the same as the Extended Link LSA for the adjacency in area 0, as shown in Example 5-59.

Example 5-65: Multi-area adjacency – Adjacency-SIDs

```
RP/0/0/CPU0:xrvr-1#show ospf 1 1 database opaque-area 8.0.0.3
self-originate

                Type-10 Opaque Link Area Link States (Area 1)

  LS age: 871
  Options: (No TOS-capability, DC)
```

```
    LS Type: Opaque Area Link
    Link State ID: 8.0.0.3
    Opaque Type: 8
    Opaque ID: 3
    Advertising Router: 1.1.1.1
    LS Seq Number: 80000001
    Checksum: 0x62ca
    Length: 68

      Extended Link TLV: Length: 44
        Link-type : 1             !! point-to-point connection
        Link ID   : 1.1.1.2       !! neighbor's router-id
        Link Data : 0.0.0.3       !! interface's ifIndex

       Adj sub-TLV: Length: 7
          Flags     : 0xe0        !! B:1,V:1,L:1,S:0
          MTID      : 0
          Weight    : 0
          Label     : 30102

       Adj sub-TLV: Length: 7
          Flags     : 0x60        !! B:0,V:1,L:1,S:0
          MTID      : 0
          Weight    : 0
          Label     : 31102

       Remote If Address sub-TLV: Length: 4
          Neighbor Address: 99.1.2.2
```

5.4.6.3 OSPFv2 LAN Adjacency-SID

The OSPF nodes on a LAN elect a Designated Router (DR) and a Backup Designated Router (BDR). The DR plays the role of the pseudonode. The DR originates the Network LSA on behalf of the network pseudonode and maintains adjacencies with all nodes on the LAN. The other nodes on the LAN, those that are not DR or BDR, are called *DROTHER* nodes. The DR and BDR election mechanism and how OSPF maintains adjacencies on a LAN is out of scope for this book.

The network topology in Figure 5-29 consists of a LAN segment with four nodes connected to it. The topology contains four nodes such that all possible adjacency types can be illustrated.

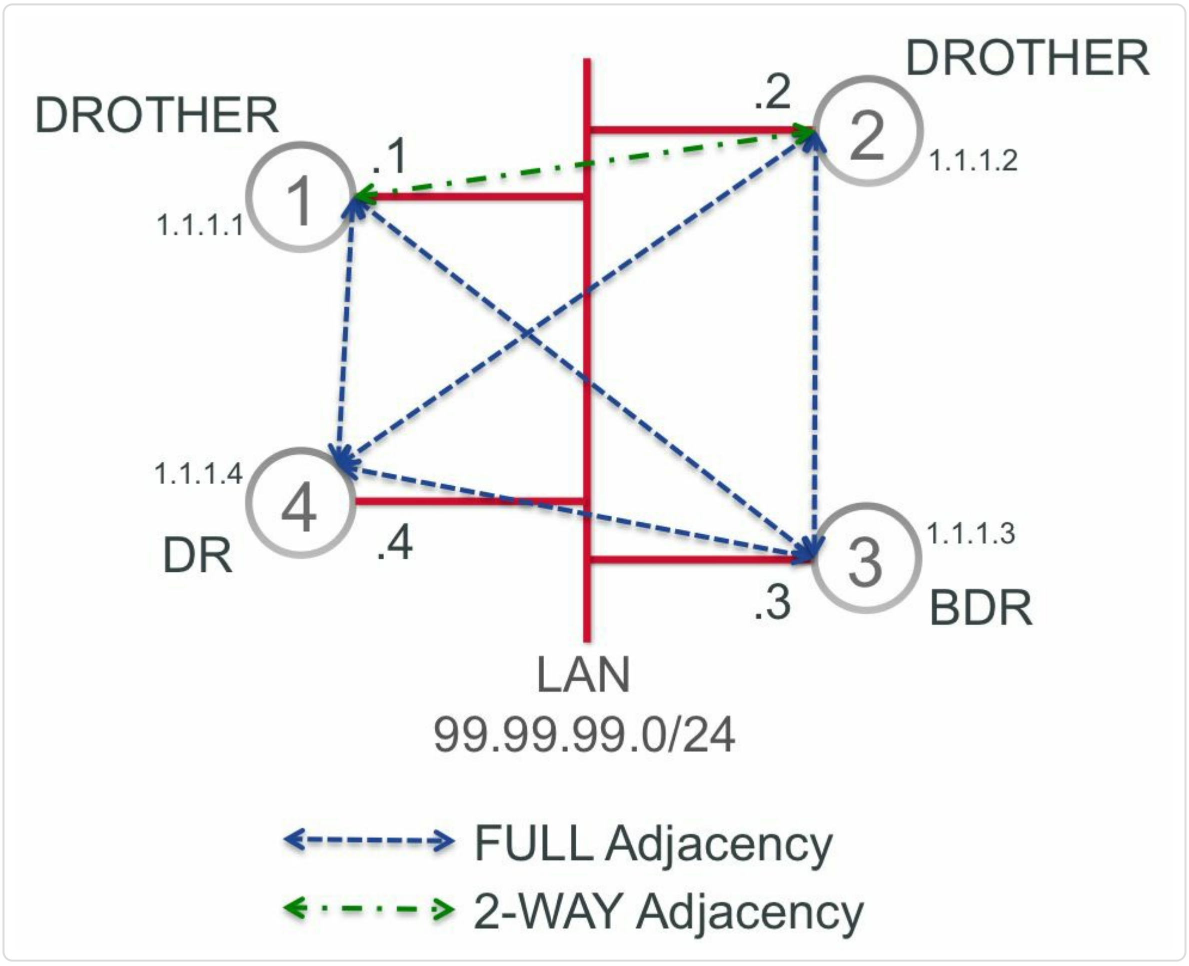


Figure 5-29: LAN with OSPF DR and BDR

In this example is the node with the highest router-id, Node4, elected as Designated Router (DR). The node with the next to highest router-id, Node3, is elected as Backup Designated Router (BDR). Node1 and Node2 are DROTHER nodes.

All OSPF nodes (DR, BDR, and DROTHER) on a LAN maintain a FULL OSPF adjacency with the DR and BDR. Each DROTHER node on a LAN, which is a node that is neither the DR nor the BDR, maintains a 2-WAY adjacency with all other DROTHER nodes on the LAN. See for example the adjacency between Node1 and Node2. An adjacency that is in 2-WAY state indicates that bi-directional communication has been established between the two nodes, both node have seen each other's hello packet.

REMINDER: OSPF Neighbor states

This is a brief summary of the different states that an OSPF adjacency with a neighbor passes through.

DOWN

No hello has been received from any neighbor.

INIT

A hello packet has been received from a neighbor, but the node's own Router ID is not included in the hello.

TWOWAY

The node's own Router ID is included in the hello, bi-directional communication has been established between the two nodes. DR and BDR are elected.

EXSTART

Master and slave roles for database exchange are determined.

EXCHANGE

Nodes exchange database descriptor (DBD) packets.

LOADING

Nodes exchange link-state information.

FULL

Nodes are fully adjacent, their databases are fully synchronized.

Example 5-66 shows the `show ospf neighbor` output as collected on the four nodes of the topology. Node3 and Node4, the BDR and DR, maintain a FULL adjacency with all other nodes on the LAN, as indicated in the third column of the output. Node1 and Node2 both maintain a FULL adjacency with DR and BDR, and maintain a 2WAY adjacency with each other.

Example 5-66: OSPF adjacencies on LAN

```
RP/0/0/CPU0:xrvr-1#show ospf neighbor
Neighbor ID     Pri   State           Dead Time
Address         Interface
1.1.1.2         1     2WAY/DROTHER    00:00:33
99.99.99.2      GigabitEthernet0/0/0/0
1.1.1.3         1     FULL/BDR        00:00:36
99.99.99.3      GigabitEthernet0/0/0/0
1.1.1.4         1     FULL/DR         00:00:36
99.99.99.4      GigabitEthernet0/0/0/0

RP/0/0/CPU0:xrvr-2#show ospf neighbor
Neighbor ID     Pri   State           Dead Time
Address         Interface
1.1.1.1         1     2WAY/DROTHER    00:00:36
99.99.99.1      GigabitEthernet0/0/0/0
1.1.1.3         1     FULL/BDR        00:00:37
99.99.99.3      GigabitEthernet0/0/0/0
1.1.1.4         1     FULL/DR         00:00:37
99.99.99.4      GigabitEthernet0/0/0/0

RP/0/0/CPU0:xrvr-3#show ospf neighbor
Neighbor ID     Pri   State           Dead Time
Address         Interface
1.1.1.1         1     FULL/DROTHER    00:00:34
99.99.99.1      GigabitEthernet0/0/0/0
1.1.1.2         1     FULL/DROTHER    00:00:32
99.99.99.2      GigabitEthernet0/0/0/0
1.1.1.4         1     FULL/DR         00:00:35
99.99.99.4      GigabitEthernet0/0/0/0

RP/0/0/CPU0:xrvr-4#show ospf neighbor
Neighbor ID     Pri   State           Dead Time
Address         Interface
1.1.1.1         1     FULL/DROTHER    00:00:39
99.99.99.1      GigabitEthernet0/0/0/0
1.1.1.2         1     FULL/DROTHER    00:00:39
99.99.99.2      GigabitEthernet0/0/0/0
1.1.1.3         1     FULL/BDR        00:00:39
```

```
  99.99.99.3      GigabitEthernet0/0/0/0
```

In the Cisco IOS XR implementation, Adjacency-SID labels are allocated for adjacencies in 2WAY state and above. The output of `show ospf neighbor detail` in Example 5-67 displays the Adjacency-SID labels for the adjacencies of Node1 to all three other nodes on the LAN (Node2, Node3, and Node4). One protection-eligible and one unprotected Adjacency-SID is allocated for each adjacency. The adjacency of Node1 to the DR (Node4) and to the BDR (Node3) is in the state FULL. The adjacency of Node1 to the DROTHER node (Node2) stays in the state 2WAY.

Example 5-67: Detailed OSPF adjacencies on LAN

```
RP/0/0/CPU0:xrvr-1#show ospf neighbor detail

* Indicates MADJ interface
# Indicates Neighbor awaiting BFD session up

Neighbors for OSPF 1

 Neighbor 1.1.1.2, interface address 99.99.99.2
    In the area 0 via interface GigabitEthernet0/0/0/0
    Neighbor priority is 1, State is 2WAY, 2 state changes
    DR is 99.99.99.4 BDR is 99.99.99.3
    Options is 0
    Dead timer due in 00:00:32
    Neighbor is up for 13:49:07
    Number of DBD retrans during last exchange 0
    Index 0/0, retransmission queue length 0, number of
retransmission 0
    First 0(0)/0(0) Next 0(0)/0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
    LS Ack list: NSR-sync pending 0, high water mark 0
    Adjacency SID Label: 30102
    Unprotected Adjacency SID Label: 31102

 Neighbor 1.1.1.3, interface address 99.99.99.3
```

```
    In the area 0 via interface GigabitEthernet0/0/0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 99.99.99.4 BDR is 99.99.99.3
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:00:33
    Neighbor is up for 13:49:01
    Number of DBD retrans during last exchange 0
    Index 1/1, retransmission queue length 0, number of
retransmission 0
    First 0(0)/0(0) Next 0(0)/0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
    LS Ack list: NSR-sync pending 0, high water mark 0
    Adjacency SID Label: 30103
    Unprotected Adjacency SID Label: 31103

 Neighbor 1.1.1.4, interface address 99.99.99.4
    In the area 0 via interface GigabitEthernet0/0/0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 99.99.99.4 BDR is 99.99.99.3
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:00:33
    Neighbor is up for 13:49:20
    Number of DBD retrans during last exchange 0
    Index 2/2, retransmission queue length 0, number of
retransmission 4
    First 0(0)/0(0) Next 0(0)/0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
    LS Ack list: NSR-sync pending 0, high water mark 0
    Adjacency SID Label: 30104
    Unprotected Adjacency SID Label: 31104


Total neighbor count: 3
```

The Adjacency-SID labels are programmed in the MPLS forwarding table of Node1, as shown in Example 5-68. All Adjacency-SIDs have the same outgoing interface but a different nexthop, towards each of the other nodes on the LAN. In the topology, NodeX has nexthop address 99.99.99.X. The output of the command shows `(idx 0)` for all entries. This is because OSPF does not use the “index” field to differentiate the Adjacency-SIDs of the same adjacency.

Example 5-68: LAN Adjacency-SID MPLS forwarding entries

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 30000 40000
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                        Switched
------ ----------- ------------------ ------------ ------------
--- ------------
30104  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.99.99.4      0
31104  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.99.99.4      0
30102  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.99.99.2      0
31102  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.99.99.2      0
30103  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.99.99.3      0
31103  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.99.99.3      0
```

Example 5-69 shows the LSD label table on Node1. The detailed output includes the label context of each label.

Example 5-69: OSPF LAN Adjacency-SIDs – MPLS label table

```
RP/0/0/CPU0:xrvr-1#show mpls label table detail
Table Label    Owner                            State  Rewrite
----- -------- -------------------------------- ------ -------
0     0        LSD(A)                           InUse  Yes
0     1        LSD(A)                           InUse  Yes
0     2        LSD(A)                           InUse  Yes
0     13       LSD(A)                           InUse  Yes
0     16000    OSPF(A):ospf-1                   InUse  No
```

```
  (Lbl-blk SRGB, vers:0, (start_label=16000, size=8000)
0      30104    OSPF(A):ospf-1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=1,
intf=Gi0/0/0/0, nh=99.99.99.4)
0      31104    OSPF(A):ospf-1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=2,
intf=Gi0/0/0/0, nh=99.99.99.4)
0      30102    OSPF(A):ospf-1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=1,
intf=Gi0/0/0/0, nh=99.99.99.2)
0      31102    OSPF(A):ospf-1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=2,
intf=Gi0/0/0/0, nh=99.99.99.2)
0      30103    OSPF(A):ospf-1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=1,
intf=Gi0/0/0/0, nh=99.99.99.3)
0      31103    OSPF(A):ospf-1                      InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=0, type=2,
intf=Gi0/0/0/0, nh=99.99.99.3)
```

5.4.6.4 OSPF LAN Adjacency-SID Advertisement

The Designated Router (DR) plays the role of the pseudonode, therefore the DR advertises a Network LSA (Type 2 LSA) on behalf of the pseudonode. This LSA contains the adjacencies of the pseudonode to all the nodes on the LAN. Example 5-70 displays the Network LSA as advertised by DR Node4 in the example topology of Figure 5-29. The pseudonode advertises an adjacency to each of the four nodes on the LAN.

Example 5-70: OSPF DR (pseudonode) advertisement

```
RP/0/0/CPU0:xrvr-4#sh ospf data network self

            OSPF Router with ID (1.1.1.4) (Process ID 1)

                Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 618
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 99.99.99.4 (address of Designated Router)
  Advertising Router: 1.1.1.4
  LS Seq Number: 80000001
  Checksum: 0xba2c
  Length: 40
  Network Mask: /24
        Attached Router: 1.1.1.1
        Attached Router: 1.1.1.2
        Attached Router: 1.1.1.3
        Attached Router: 1.1.1.4
```

Each node on the LAN, including DR and BDR, then advertises its own adjacency to this pseudonode. Example 5-71 displays the Router LSA (Type 1 LSA) as advertised by Node1. It indicates Node1 is connected to a Transit Network with DR Node4, this is the LAN segment that is represented by the pseudonode, which is then represented by DR Node4.

Example 5-71: OSPF adjacency to DR (pseudonode) advertisement

```
RP/0/0/CPU0:xrvr-1#show ospf data router self-originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 0)

  LS age: 290
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.1.1.1
  Advertising Router: 1.1.1.1
  LS Seq Number: 8000001e
  Checksum: 0xbae4
  Length: 48
   Number of Links: 2

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 1.1.1.1
     (Link Data) Network Mask: 255.255.255.255
      Number of TOS metrics: 0
```

```
          TOS 0 Metrics: 1

        Link connected to: a Transit Network
         (Link ID) Designated Router address: 99.99.99.4
         (Link Data) Router Interface address: 99.99.99.1
          Number of TOS metrics: 0
           TOS 0 Metrics: 1
```

OSPF Adjacency-SIDs for adjacencies on a LAN interface are advertised in Extended Link Opaque LSAs, the same as for point-to-point Adjacency-SIDs. The fields of the Extended Link TLV that is included in the Extended Link LSA, identify the adjacency that this TLV applies to. The Adjacency-SIDs are added to this Extended Link TLV as a combination of Adjacency-SID sub-TLVs and LAN Adjacency-SID Sub-TLVs. The Extended Link TLV for a LAN adjacency contains Adjacency-SID sub-TLVs for the adjacency to the DR and LAN Adjacency sub-TLVs for the adjacencies to the other, non-DR nodes on the LAN, including the BDR.

The format of a LAN Adjacency-SID sub-TLV is shown in Figure 5-30.

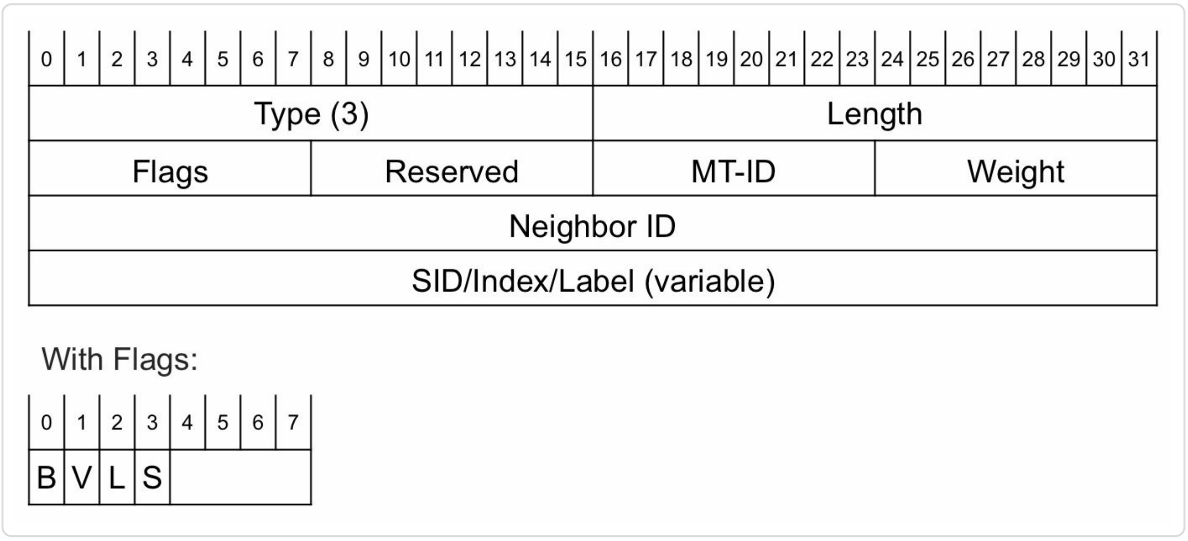


Figure 5-30: LAN Adjacency-SID sub-TLV format

The LAN Adjacency-SID sub-TLV contains the following fields:

- The Flags, MT-ID, and Weight fields are identical to the fields in the Adjacency-SID sub-TLV. Please refer to section 5.4.6.1 "OSPFv2 Adjacency-SID Advertisement" for these fields.
- Neighbor ID: router-id of the neighbor node
- SID/Index/Label: with V-flag=1 and L-flag=1: the 20 rightmost bits are used to encode a MPLS label value – always a MPLS label value (V:1, L:1) in IOS XR

In the simple topology of Figure 5-29, Node1 only has a single adjacency and consequently only advertises a single Extended Link LSA. The Link State ID of that LSA is 8.0.0.4 in this example, Opaque-Type 8 and Opaque-ID 4. This LSA is shown in Example 5-72.

The Extended Link LSA contains an Extended Link TLV. The fields in that TLV (lines 18-21) identify the adjacency that this TLV applies to. The Link-type is 2. This means it is a "Connection to a Transit Network", which is a connection to a LAN pseudonode. Link ID and Link Data then indicate the Designated Router address (99.99.99.4) and Router Interface address (99.99.99.1). These three fields match the corresponding fields of the pseudonode adjacency in the Router LSA of Node1 in Example 5-71. This is the adjacency that this Extended Link TLV applies to.

The Extended Link TLV contains six sub-TLVs, each containing one Adjacency-SID: three pairs of Adjacency-SIDs (one protection-eligible and one unprotected), one pair for each of the other nodes on the LAN.

The first sub-TLVs in the LSA (lines 23-49) are the LAN Adjacency-SID sub-TLVs. These contain the Adjacency-SIDs for the adjacencies to the non-DR nodes (Node2 and Node3). Note that the Adjacency-SID for the adjacency to the BDR is also advertised using a LAN Adjacency-SID sub-TLV, even though Node1 maintains a FULL adjacency with the BDR. The neighbor ID fields of these sub-TLVs contain the router-id of the remote node to identify the remote node for the Adjacency-SID. For example the first LAN Adjacency-SID (lines 23-28) applies to the adjacency with Node2. Note that two Adjacency-SIDs are advertised to each remote node, one protection-eligible (B:1) and one unprotected (B:0).

The last two sub-TLVs in the TLV (lines 51-61) are the Adjacency-SID sub-TLVs for the adjacency to the DR Node4.

Example 5-72: OSPF adjacency to pseudonode with LAN Adjacency-SIDs

```
 1| RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 8.0.0.4
self-originate
 2|
 3|            OSPF Router with ID (1.1.1.1) (Process ID 1)
 4|
 5|                Type-10 Opaque Link Area Link States (Area
0)
 6|
 7|   LS age: 1154
 8|   Options: (No TOS-capability, DC)
 9|   LS Type: Opaque Area Link
10|   Link State ID: 8.0.0.4
11|   Opaque Type: 8
12|   Opaque ID: 4
13|   Advertising Router: 1.1.1.1
14|   LS Seq Number: 8000001e
15|   Checksum: 0xcb2e
16|   Length: 124
17|
18|     Extended Link TLV: Length: 100
19|       Link-type : 2           !! Connection to a Transit
Network
20|       Link ID   : 99.99.99.4  !! Designated Router address
21|       Link Data : 99.99.99.1  !! Router Interface address
22|
      LAN Adj sub-TLV: Length: 11
        Flags      : 0xe0          !! B:1,V:1,L:1,S:0
        MTID       : 0
        Weight     : 0
        Neighbor ID: 1.1.1.2
        Label     : 30102

      LAN Adj sub-TLV: Length: 11
        Flags      : 0x60          !! B:0,V:1,L:1,S:0
        MTID       : 0
        Weight     : 0
        Neighbor ID: 1.1.1.2
        Label     : 31102

      LAN Adj sub-TLV: Length: 11
        Flags      : 0xe0          !! B:1,V:1,L:1,S:0
        MTID       : 0
        Weight     : 0
        Neighbor ID: 1.1.1.3
        Label     : 30103

      LAN Adj sub-TLV: Length: 11
        Flags      : 0x60          !! B:0,V:1,L:1,S:0
        MTID       : 0
        Weight     : 0
        Neighbor ID: 1.1.1.3
        Label     : 31103

      Adj sub-TLV: Length: 7
         Flags     : 0xe0          !! B:1,V:1,L:1,S:0
         MTID      : 0
         Weight    : 0
         Label     : 30104

      Adj sub-TLV: Length: 7
         Flags     : 0x60          !! B:0,V:1,L:1,S:0
         MTID      : 0
         Weight    : 0
         Label     : 31104
```

5.5 IGP Multi-Area/Level Operations

Enabling Segment Routing in a network does not change how multi-area and multi-level functionality works for ISIS or OSPF. There is no change required in existing IGP network designs.

When a prefix is propagated between (OSPF) areas or (ISIS) levels, the Prefix-SID that is associated with that prefix is propagated as well, together with its associated prefix. In other words, the Prefix-SID information stays attached or associated with the prefix that is propagated between areas or levels. This is possible since the Prefix-SID is unique within the whole Segment Routing Domain, which crosses the area boundaries.

Adjacency-SIDs are not propagated between areas or levels since they are attached to links, and links themselves are not propagated between areas or levels by the base IGP.

To propagate Prefix-SIDs with their prefixes, the ISIS L1L2 node or OSPF Area Border Router (ABR) must be Segment Routing enabled.

5.5.1 Determining Penultimate Hop for a Prefix

In the LDP or BGP Labeled Unicast world, for a node to find out it is the penultimate hop node of a Label Switched Path is straightforward: it does not need to find this out! The behavior of the penultimate hop is specified by the ultimate hop that advertises an attached or connected prefix with an implicit-null (or explicit-null or the actual) label to the penultimate hop node. This way, the penultimate hop automatically applies the behavior as required by the originating node; it simply needs to install the received label in the forwarding table. It works like this because of the hop-by-hop signaling of the label binding.

This mechanism does not apply to IGP Segment Routing (but it does for BGP Segment Routing, see chapter 6, "Segment Routing BGP Control Plane"). Segment Routing does not use hop-by-hop signaling; it advertises its SIDs domain-wide. All nodes in the area receive the same Segment Routing information through the link state advertisements. The node that originates a Prefix-SID, signals the required behavior of the penultimate hop for that Prefix Segment by setting or clearing flags (i.e. PHP-off and explicit-null) that are included in the Prefix-SID advertisement. Each node must then figure out itself if it is the penultimate hop for a Prefix Segment. If it is the penultimate hop then it must apply the behavior as indicated by the Prefix-SID flags. If it is not, then it should not apply any penultimate hop operation.

For a node to know it is the penultimate hop, it needs two pieces of information:

- Which node is the downstream neighbor for a Prefix Segment? And,
- Is the downstream neighbor connected to the prefix associated with the Prefix Segment?

The first item is straightforward: the downstream neighbor for a Prefix Segment is the nexthop to reach the prefix of that Prefix Segment. This is derived from the Shortest Path Tree.

The second item is not that simple. One may think that the node advertising the prefix is also connected to the prefix, but these simple assumptions are not always correct when we look at the multi-area/level operations.

The multi-area functionality differs between the two link-state protocols ISIS and OSPF.

5.5.1.1 ISIS Determination of Penultimate Hop with Multi-Level

The ISIS multi-area functionality as specified in IETF RFC 1195 propagates by default all Level-1 prefixes into Level-2, but no prefix propagation was allowed in the other direction, from Level-2 into Level-1. Level-1 nodes had to rely on discovering L1L2 nodes that are attached to another area. These L1L2 nodes have the ATT-bit (Attach) set in their L1 Link State PDU (LSP). L1 nodes send all their inter-area traffic to the nearest L1L2 node(s) with the ATT-bit set in their LSP. IETF RFC 5302 introduced prefix propagation, or prefix leaking, from Level-2 into Level-1. To prevent routing loops, the Up/Down-bit was introduced to indicate if a prefix was propagated downwards, from Level-2 to Level-1. Such prefix has the Up/Down-bit set. If that Up/Down-bit is set, then it is not allowed to propagate that prefix from Level-1 to Level-2 again since that could lead to a routing loop. This Up/Down-bit is the only piece of information that indicates if a prefix has been propagated between levels, and only for the L2→L1 direction. L1→L2 inter-area routes are indistinguishable from L2 intra-area prefixes. The same is the case for prefixes redistributed into ISIS, they are also indistinguishable from intra-area prefixes. So, a node can determine from the LSP the node that advertised a prefix, but it cannot determine which node originated the prefix, to which node the prefix is connected.

It is clear that more information is needed to determine if a prefix is connected to a node. IETF RFC 7794 introduces additional ISIS prefix attributes that enable determination if a node originated a prefix or not. This RFC introduces the generic prefix attribute X-flag (External Prefix), R-flag (Re-advertised), and N-flag (Node). The use of R- and N-flags are similar to the flags in the Prefix-SID with the same name. The R-flag is set when the prefix is propagated between levels, upwards or downwards. The N-flag has nothing to do with this section but is added here for completeness. It is the Node flag, indicating if the prefix identifies the node that advertises it. The X-flag has no equivalent in the Prefix-SID. It is set when the prefix has been redistributed into ISIS.

If a node advertises a prefix that has both the prefix attribute X- and R-flags not set, then that prefix is local to the node advertising the prefix and the upstream neighbors are penultimate hop nodes for that prefix.

5.5.1.2 OSPF Determination of Penultimate Hop with Multi-Area

It is simpler for OSPF. Intra-area prefixes are distributed using Router LSAs. If a prefix is advertised in a Router LSA, it is by definition a prefix local to the area, it cannot be a propagated prefix. The node that originates the Router LSA that advertises reachability to a prefix is connected to that prefix. Each node can therefore determine if it is the penultimate hop of a Prefix Segment associated with such prefix.

For inter-area prefixes, prefixes advertised in Summary LSAs, one could (wrongly) assume that the Area Border Router (ABR) that advertises such prefix is not connected to that prefix but that the ABR propagated a remote prefix into the local area. That assumption is not always correct. OSPF can only originate each prefix in a single area. This includes its own, locally connected prefixes. So, even if a prefix is local to the ABR, it is propagated by the ABR to the other connected areas and advertised as inter-area prefix in those areas. E.g. a loopback prefix 1.1.1.1/32 configured in area 0 is advertised as intra-area prefix (in Router LSA) in area 0 and as inter-area prefix (in Summary LSA) in the other areas connected to the ABR, e.g. areas 1 and 2. If a node in area 1, receiving the inter-area advertisement, would conclude that 1.1.1.1/32 is not connected to the ABR since it is an inter-area prefix then that would be wrong. Prefix 1.1.1.1/32 is connected to the ABR, but in a different area.

More information is needed to determine if an inter-area prefix is local to the ABR or not. IETF RFC 7684 specifies the A-flag (Attach) of the Extended Prefix TLV for this purpose. This A-flag is set if the inter-area prefix, that this Extended Prefix TLV applies to, is locally connected or attached in another connected area of the ABR.

5.5.2 Propagation of Prefix-SID Across Areas/Levels

Some flags in the Prefix-SID information are updated by the border node when propagating it between areas or levels to ensure that when a packet arrives at it from the source node, the correct MPLS label is on the top of the packet label stack for the node to forward it correctly across the area/level border if required. These flags are the ones that specify the behavior required from the penultimate node: PHP-off flag and Explicit-null flag. ISIS also updates the Re-advertisement flag.

When an OSPF Area Border Router or ISIS Level-1/Level-2 router propagates a prefix with its associated Prefix-SID from one area or level to another, in general it updates the Prefix-SID flags as follows:

- **Set** the PHP-off flag (ISIS: P-flag, OSPF: NP-flag) of the prefix-SID
- **Clear** the Explicit-null flag (E-flag) of the prefix-SID
- IS-IS also **sets** the Re-advertisement flag (R-flag) for all prefixes that are propagated between levels. OSPF does not need such flag since for OSPF it is evident from the type of LSA if a prefix has been propagated.

The flags are updated as indicated above, except if the prefix/Prefix-SID that is propagated is directly connected or attached to the propagating border node (L1L2, ABR). In that case this border node does not change the PHP-off flag, Explicit-null flag, and ISIS Re-advertisement flag. But in that case

OSPF sets the Attached flag (A-flag) of the Extended Prefix TLV (see section 5.3.4 “Prefix-SID Advertisement in OSPFv2”).

The ISIS Prefix-SID R-flag really is a prefix property that is independent from Segment Routing. Therefore the function of the Prefix-SID R-flag will likely be taken over by the R-flag and X-flag in the generic ISIS prefix attributes specified in IETF RFC 7794.

Once these flags have been setup correctly by the border router, all nodes can simply apply the correct PHP behavior by only determining the *advertising* node of the Prefix-SID advertisement from its perspective without bothering about area/level boundaries. Each node only has to determine if it is an upstream neighbor of the node *advertising* the Prefix-SID. This simplifies operations in general for all nodes and for all types of prefixes.

Figure 5-31 illustrates the propagation of prefixes between areas in an OSPF network. The illustration is equally applicable to a multi-level ISIS network. The network topology consists of three nodes and the middle node, Node2, is the Area Border Router (ABR). The nodes advertise the prefixes with Prefix-SIDs shown in Table 5-3. Node2 advertises two prefixes, one in Area 0 and one in Area 1. The Prefix-SIDs advertised in Area 1 have the E-flag and NP-flag set; the Prefix-SIDs require Explicit-null behavior.

Table 5-3: Prefix propagation: prefixes and Prefix-SIDs

	prefix	**Prefix-SID**	**Original flags**	**Area**
Node1	1.0.0.1/32	16001	NP:0; E:0	0
Node2	1.0.0.2/32	16002	NP:0; E:0	0
	1.1.1.2/32	17002	NP:1; E:1	1
Node3	1.1.1.3/32	16003	NP:1; E:1	1

Table 5-4 illustrates how the Prefix-SID flags are updated by the ABR when propagating the Prefix-SIDs between areas. Since prefixes 1.0.0.2/32 and 1.1.1.2/32 are local to Node2, Node2 does not update their Prefix-SID flags when propagating between areas. Node2 sets the NP-flag (PHP-off) for Prefix-SID 16001 (prefix 1.0.0.1/32) and Prefix-SID 16003 (prefix 1.1.1.3/32). Node2 also clears the E-flag (Explicit-null) for Prefix-SID 16003.

Table 5-4: Prefix propagation, updated Prefix-SID flags

prefix	**Prefix-SID**	**Flags in Area 0**	**propagation**	**Flags in Area 1**
1.0.0.1/32	16001	NP:0; E:0	→	NP:1; E:0
1.0.0.2/32	16002	NP:0; E:0	→	NP:0; E:0

1.1.1.2/32	17002	NP:1; E:1	←	NP:1; E:1
1.1.1.3/32	16003	NP:1; E:0	←	NP:1; E:1

Figure 5-31 illustrates the labels of packets sent to the four Prefix-SIDs in the network. For example, Node2 propagates prefixes 1.1.1.2/32 and 1.1.1.3/32 from Area 1 to Area 0 and advertises them to its neighbor in Area 0: Node1. Node1 knows that the advertising node of these prefixes is its downstream neighbor Node2. Therefore, Node1 applies the behavior as indicated by the Prefix-SID flags to packets it receives on this Prefix Segment. For Prefix-SID 16002 it applies Explicit-null behavior, so it swaps 16002 with Explicit-null. For Prefix-SID 16003 it applies no-PHP behavior, so it swaps 16003 with 16003.

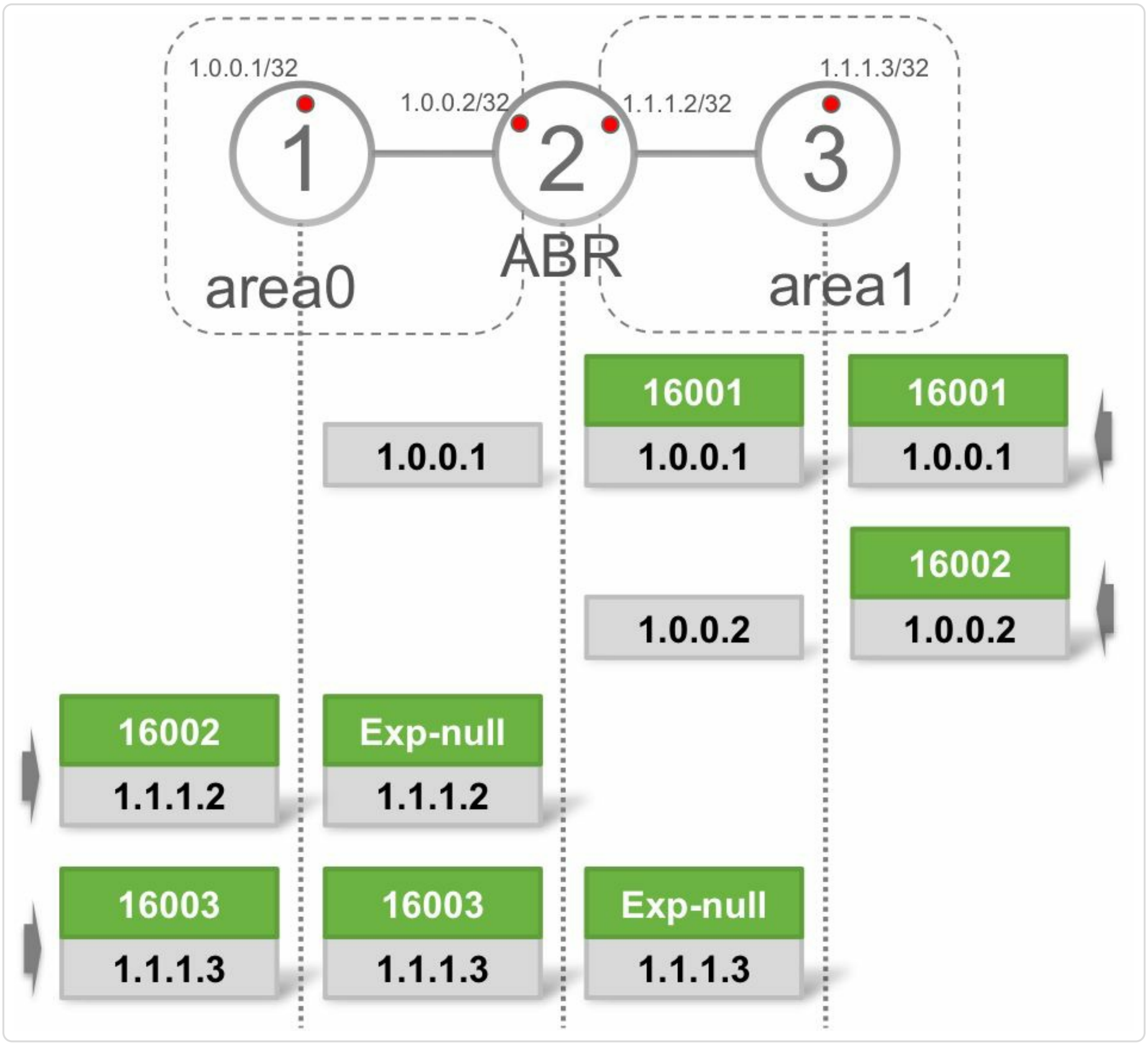


Figure 5-31: Propagation of prefixes between areas and levels

When ISIS propagates a prefix between levels, then ISIS attaches the Prefix-SID sub-TLV to the IP reachability TLV of the propagated prefix. The selection of prefixes that must be propagated is independent from Segment Routing. The Prefix-SID simply follows the prefix.

Contrary to ISIS, OSPF carries the prefix reachability and Prefix-SID information in different LSAs. OSPF uses the following procedure to propagate Prefix-SIDs between areas. When an OSPF ABR propagates an intra-area prefix or an inter-area prefix to all its other connected areas, it advertises that prefix in a Summary LSA (type 3 LSA) and it also originates an Extended Prefix LSA for that prefix in the same areas as the Summary LSA. The Extended Prefix LSA has area flooding scope (type 10 LSA), the Route-type field in the OSPF Extended Prefix TLV is set to “inter-area” and the Prefix-SID field of the included Prefix-SID Sub-TLV is set to the Prefix-SID index of the prefix. Additionally, OSPF on ABR originates this “inter-area” Extended Prefix LSA into an area, if and only if it is also originating its corresponding Summary LSA into it. When the Summary LSA is purged, its corresponding Extended Prefix LSA is also withdrawn.

Another aspect that is common to both IGPs and needs to be kept in mind is regarding summarization of prefixes that is sometimes done at the border router between area/levels. In such conditions, the individual (usually host) prefixes that have the associated Prefix-SIDs would not be propagated into the other area and instead a Summary Prefix is originated. The remote area would then not receive the individual Prefix-SIDs, but also the Summary Prefix would not have a Prefix-SID – thus resulting in no Prefix Segment being available for Segment Routing forwarding towards those remote area prefix destinations. The origination of a default route into a stub area in OSPF or the Level 1 area in ISIS is similar to summarization in that there would be no Prefix Segment corresponding to this default route in that area.

Note that summarization or default route usage, does not break IGP forwarding when used with Segment Routing – only that the forwarding is not done using Prefix Segments but using normal IP or LDP based forwarding.

HIGHLIGHT

Multi-area/level designs for IGP networks work seamlessly for Segment Routing.

Prefix-SIDs are propagated by IGPs across area/level borders along with the base prefix reachability info to enable seamless flow for Segment Routing traffic across areas/levels.

The correct Segment Routing MPLS forwarding behavior is ensured across area/level borders.

5.5.3 IS-IS Multi-Area Example

Figure 5-32 shows the topology used for the ISIS inter-area example. It is a chain of seven nodes located in three areas. Node1, Node2, and Node3 are in area 49.0001. Node4 is in area 49.0000. Node5, Node6, and Node7 are in area 49.0002.

Node1, Node2, Node6, and Node7 are Level-1 nodes. Node3 and Node5 are Level-1-2 nodes and Node4 is a Level-2-only node. Note that the Level-2 topology spans Node3, Node4, and Node5. The Level-1 topologies span Node1, Node2, and Node3; and Node5, Node6, and Node7.

Both IPv4 and IPv6 address-families are configured on all nodes.

The L1L2 nodes are configured to propagate all host prefixes between levels, both from Level-1 to Level-2 and from Level-2 to Level-1.

The L1L2 nodes have two loopback prefixes, one in each level.

All nodes allocated the same SRGB [16000-23999].

Prefix and Prefix-SID conventions:

- The Loopback IPv4 prefix of node N in level L: 1.L.1.N/32.
- The Loopback IPv6 prefix of node N in level L: 2001::1:L:1:N/128.
- IPv4 Prefix-SID of node N in level L: 16000 + 100*L + N
- IPv6 Prefix-SID of node N in level L: 17000 + 100*L + N

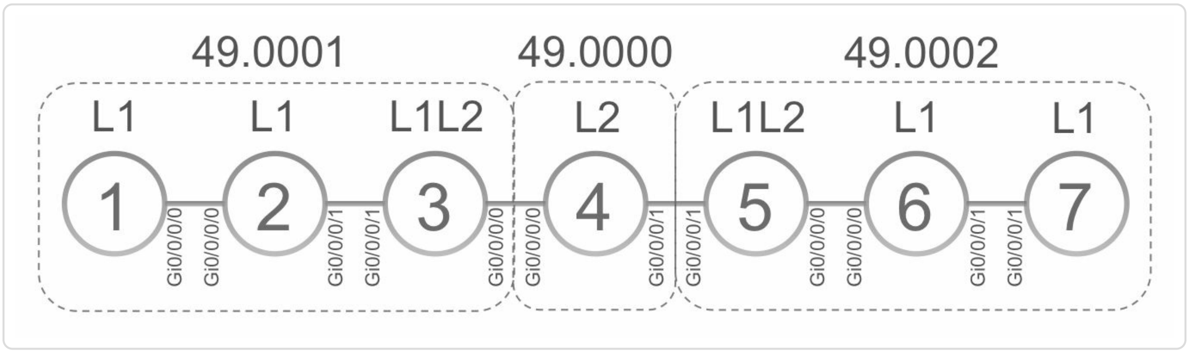


Figure 5-32: ISIS multi-area, multi-level topology

Figure 5-33 and Figure 5-34 show the loopback prefix and associated prefix-SID assignments for each node in each area, for IPv4 and IPv6 topologies respectively. The column indicates the node and the row indicates the area and level. The cell content specifies the loopback prefix and its Prefix-SID as configured for that (node, area, level). For example Node3 has two loopbacks, one in Level-1 (1.1.1.3/32, SID 16103 and 2001::1:1:1:3/128, SID 17103) and one in Level-2 (1.2.1.3/32, SID 16203 and 2001::1:2:1:3/128, SID 17203).

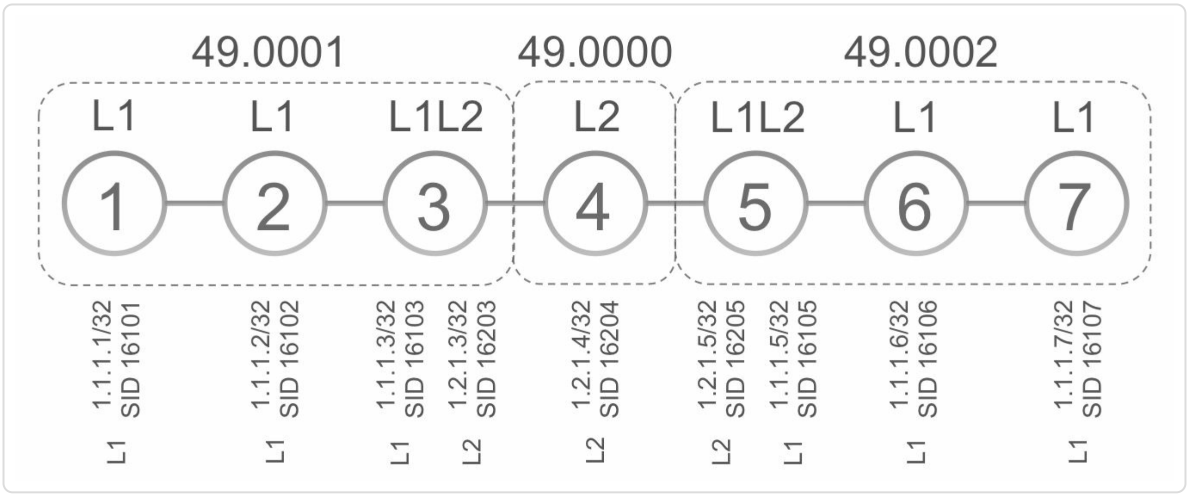


Figure 5-33: ISIS multi-area – IPv4 loopback prefixes and Prefix-SIDs

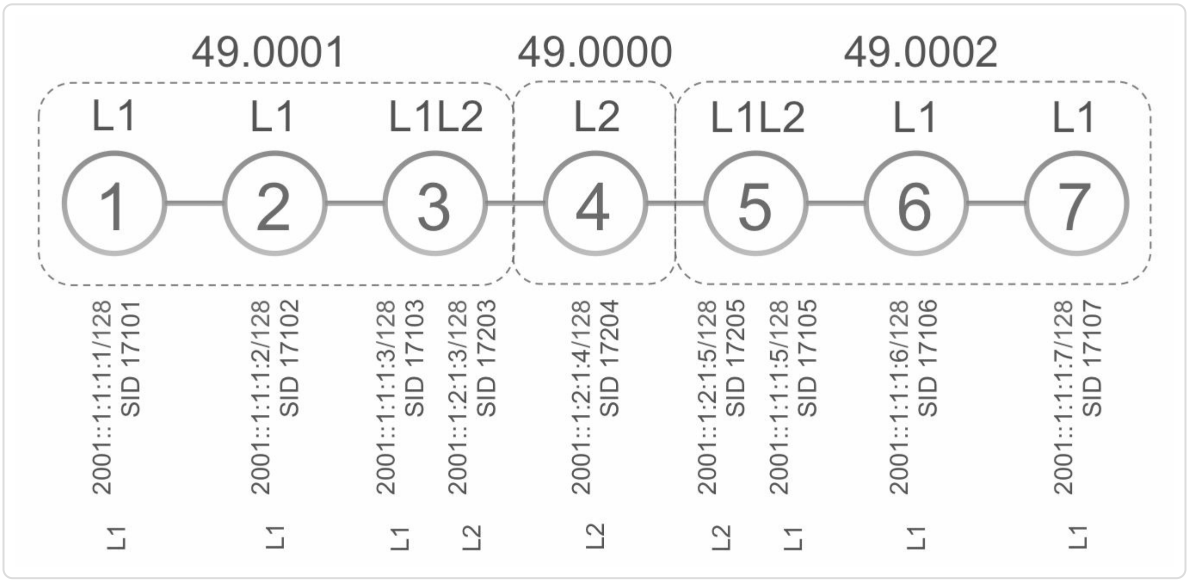

49.0001
49.0000
49.0002
L1
L1
L1L2
L2
L1L2
L1
L1
1
2
3
4
5
6
7
L1 2001::1:1:1:1/128 SID 17101
L1 2001::1:1:1:2/128 SID 17102
L1 2001::1:1:1:3/128 SID 17103
L2 2001::1:2:1:3/128 SID 17203
L2 2001::1:2:1:4/128 SID 17204
L2 2001::1:2:1:5/128 SID 17205
L1 2001::1:1:1:5/128 SID 17105
L1 2001::1:1:1:6/128 SID 17106
L1 2001::1:1:1:7/128 SID 17107

Figure 5-34: ISIS multi-area – IPv6 loopback prefixes and Prefix-SIDs

An IPv4 traceroute from Node7 to Node1 shows there is end-to-end inter-area connectivity using the same Prefix-SID label 16101. See Example 5-73.

Example 5-73: ISIS multi-area – IPv4 traceroute

```
RP/0/0/CPU0:xrvr-7#traceroute 1.1.1.1 source 1.1.1.7

Type escape sequence to abort.
Tracing the route to 1.1.1.1
```

```
 1  99.6.7.6 [MPLS: Label 16101 Exp 0] 219 msec  29 msec  119
msec
 2  99.5.6.5 [MPLS: Label 16101 Exp 0] 29 msec  39 msec  39
msec
 3  99.4.5.4 [MPLS: Label 16101 Exp 0] 39 msec  39 msec  59
msec
 4  99.3.4.3 [MPLS: Label 16101 Exp 0] 39 msec  49 msec  39
msec
 5  99.2.3.2 [MPLS: Label 16101 Exp 0] 49 msec  49 msec  29
msec
 6  99.1.2.1 39 msec  29 msec  29 msec
```

An IPv6 traceroute from Node7 to Node1 shows the output as displayed in Example 5-74.

Example 5-74: ISIS multi-area – IPv6 traceroute

```
RP/0/0/CPU0:xrvr-7#traceroute ipv6 2001::1:1:1:1 source
2001::1:1:1:7

Type escape sequence to abort.
Tracing the route to 2001::1:1:1:1

 1  2001::99:6:7:6 [MPLS: Label 17101 Exp 0] 59 msec 39 msec 39
msec
 2  2001::99:5:6:5 [MPLS: Label 17101 Exp 0] 59 msec 29 msec 39
msec
 3  2001::99:4:5:4 [MPLS: Label 17101 Exp 0] 49 msec 29 msec 39
msec
 4  2001::99:3:4:3 [MPLS: Label 17101 Exp 0] 49 msec 39 msec 49
msec
 5  2001::99:1:3:2 [MPLS: Label 17101 Exp 0] 59 msec 29 msec 39
msec
 6  2001::1:1:1:1 299 msec 39 msec 49 msec
```

This example follows the propagation of prefixes from left (Node1) to right (Node7). In particular the prefixes advertised by Node1, Node3 and Node5 are tracked when they propagate in the three areas.

The Node1 ISIS configuration is displayed in Example 5-75. Node1 is configured as a Level-1 only node.

Example 5-75: ISIS multi-area – ISIS configuration of Node1

```
interface Loopback0
 ipv4 address 1.1.1.1/32
 ipv6 address 2001::1:1:1:1/128
!
router isis 1
 is-type level-1
 net 49.0001.0000.0000.0001.00
 address-family ipv4 unicast
  metric-style wide
  segment-routing mpls
 !
 address-family ipv6 unicast
  metric-style wide
  segment-routing mpls
 !
 interface Loopback0
  passive
  circuit-type level-1
  address-family ipv4 unicast
   prefix-sid absolute 16101
  !
  address-family ipv6 unicast
   prefix-sid absolute 17101
  !
 !
 interface GigabitEthernet0/0/0/0
  circuit-type level-1
  point-to-point
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
 !
```

Example 5-76 shows the ISIS L1 LSP advertised by Node1 in area 49.0001. Node1 is a Level-1 node that advertises IPv4 prefix 1.1.1.1/32 with its associated Prefix-SID index 101 (absolute 16101) and IPv6 prefix 2001::1:1:1:1/128 with its associated Prefix-SID index 4101 (absolute 17101). For both Prefix-SIDs Node1 requests default PHP behavior: the P-flag (PHP-off) is unset. Also the Prefix-SID R-flag (Re-advertised) is unset since these prefixes have not been propagated between levels.

Example 5-76: ISIS multi-area – L1 LSP of Node1

```
RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-1 level 1

IS-IS 1 (Level-1) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x00000013   0x1cce
1170           0/0/0
  Area Address:   49.0001
  NLPID:          0xcc
  NLPID:          0x8e
  MT:             Standard (IPv4 Unicast)
  MT:             IPv6 Unicast
0/0/0
  Hostname:       xrvr-1
  IP Address:     1.1.1.1
  IPv6 Address:   2001::1:1:1:1
  Router Cap:     1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 10         IS-Extended xrvr-2.00
    Interface IP Address: 99.1.2.1
    Neighbor IP Address: 99.1.2.2
    ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:30102
    ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:31102
!! R-flag and P-flag are not set for prefix-SID of prefix
1.1.1.1/32
  Metric: 0          IP-Extended 1.1.1.1/32
    Prefix-SID Index: 101, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 10         IP-Extended 99.1.2.0/24
  Metric: 10         MT (IPv6 Unicast) IS-Extended xrvr-2.00
    Interface IPv6 Address: 2001::99:1:2:1
    Neighbor IPv6 Address: 2001::99:1:2:2
    ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:32102
    ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:33102
```

```
!! R-flag and P-flag are not set for prefix-SID of prefix
2001::1:1:1:1/128
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
    Prefix-SID Index: 1101, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 10         MT (IPv6 Unicast) IPv6 2001::99:1:2:0/112

 Total Level-1 LSP count: 1     Local Level-1 LSP count: 1
```

Node3 is a Level-1-2 node. By default all interfaces are in both Levels. But each of the interfaces of Node3 is configured in a specific level using the `circuit-type` interface configuration. Interfaces Loopback0 and Gi0/0/0/0 are only in Level-2. Interfaces Loopback1 and Gi0/0/0/1 are only in Level-1.

By default, all prefixes are leaked from Level-1 to level-2, but Node3 is configured to only leak the host prefixes. And Node3 is also configured to propagate host prefixes from Level-2 to Level-1. A route-policy is used to specify which prefixes are propagated between levels. Example 5-77 displays the configuration of Node3. The prefix-set `(0.0.0.0/0 eq 32)` in the route-policy matches all /32 IPv4 prefixes.

Example 5-77: ISIS multi-area – ISIS configuration of Node3

```
interface Loopback0
 description Loopback
 ipv4 address 1.2.1.3 255.255.255.255
 ipv6 address 2001::1:2:1:3/128
!
interface Loopback1
 ipv4 address 1.1.1.3 255.255.255.255
 ipv6 address 2001::1:1:1:3/128
!
route-policy LOOPBACKS
  if destination in (0.0.0.0/0 eq 32) or destination in (::/0
eq 128) then
    pass
  else
    drop
  endif
end-policy
!
router isis 1
 net 49.0001.0000.0000.0003.00
 address-family ipv4 unicast
  metric-style wide
  propagate level 1 into level 2 route-policy LOOPBACKS
  propagate level 2 into level 1 route-policy LOOPBACKS
  segment-routing mpls
 !
 address-family ipv6 unicast
  metric-style wide
  propagate level 1 into level 2 route-policy LOOPBACKS
  propagate level 2 into level 1 route-policy LOOPBACKS
  segment-routing mpls
 !
 interface Loopback0
  passive
  circuit-type level-2-only
  address-family ipv4 unicast
   prefix-sid absolute 16203
  !
  address-family ipv6 unicast
   prefix-sid absolute 17203
  !
 !
 interface Loopback1
  passive
  circuit-type level-1
  address-family ipv4 unicast
   prefix-sid absolute 16103
  !
  address-family ipv6 unicast
   prefix-sid absolute 17103
  !
 !
 interface GigabitEthernet0/0/0/0
  circuit-type level-2-only
  point-to-point
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
 !
 interface GigabitEthernet0/0/0/1
  circuit-type level-1
  point-to-point
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
 !
```

Example 5-78 shows the ISIS Level-2 LSP as advertised by Node3 into the Level-2 backbone area. Node3 propagated the Level-1 prefixes 1.1.1.1/32 and 2001::1:1:1:1/128, both originated by Node1, from Level-1 to Level-2. The flags of their associated Prefix-SIDs are updated: the P-flag (PHP-off) flag and the R-flag (Re-advertised) are both set.

The IPv4 prefix 1.1.1.3/32 and IPv6 prefix 2001::1:1:1:3/128 are Level-1 prefixes that are local to Node3. Even though these prefixes are also propagated from Level-1 to Level-2, the P-flag (PHP-off) and R-flag (Re-advertised) of their Prefix-SIDs have not been set, since these prefixes are local to the L1L2 node.

Example 5-78: ISIS multi-area – L2 LSP advertised by Node3

```
RP/0/0/CPU0:xrvr-3#show isis database verbose xrvr-3 level 2

IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-3.00-00        * 0x0000001a   0xfb50
1099           0/0/0
  Area Address:   49.0001
  NLPID:          0xcc
  NLPID:          0x8e
  MT:             Standard (IPv4 Unicast)
  MT:             IPv6 Unicast
```

```
0/0/0
  Hostname:       xrvr-3
  IP Address:     1.2.1.3
  IPv6 Address:   2001::1:2:1:3
  Router Cap:     1.2.1.3, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 10         IS-Extended xrvr-4.00
    Interface IP Address: 99.3.4.3
    Neighbor IP Address: 99.3.4.4
    ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:30304
    ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:31304
!! R-flag and P-flag are set for prefix-SID of prefix
1.1.1.1/32
  Metric: 20         IP-Extended 1.1.1.1/32
    Prefix-SID Index: 101, Algorithm:0, R:1 N:1 P:1 E:0 V:0 L:0
  Metric: 10         IP-Extended 1.1.1.2/32
    Prefix-SID Index: 102, Algorithm:0, R:1 N:1 P:1 E:0 V:0 L:0
!! R-flag and P-flag are not set for prefix-SID of prefix
1.1.1.3/32
!! prefix 1.1.1.3/32 has been propagated from L1 to L2, but is
local to Node3
  Metric: 0          IP-Extended 1.1.1.3/32
    Prefix-SID Index: 103, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 0          IP-Extended 1.2.1.3/32
    Prefix-SID Index: 203, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 10         IP-Extended 99.3.4.0/24
  Metric: 10         MT (IPv6 Unicast) IS-Extended xrvr-4.00
    Interface IPv6 Address: 2001::99:3:4:3
    Neighbor IPv6 Address: 2001::99:3:4:4
    ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:32304
    ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:33304
!! R-flag and P-flag are set for prefix-SID of prefix
2001::1:1:1:1/128
  Metric: 20         MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128
    Prefix-SID Index: 1101, Algorithm:0, R:1 N:1 P:1 E:0 V:0
L:0
  Metric: 10         MT (IPv6 Unicast) IPv6 2001::1:1:1:2/128
    Prefix-SID Index: 1102, Algorithm:0, R:1 N:1 P:1 E:0 V:0
L:0
!! R-flag and P-flag are not set for prefix-SID of prefix
2001::1:1:1:3/128
!! prefix 2001::1:1:1:3/128 has been propagated from L1 to L2,
but is local to Node3
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:3/128
```

```
    Prefix-SID Index: 1103, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:2:1:3/128
    Prefix-SID Index: 1203, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 10         MT (IPv6 Unicast) IPv6 2001::99:3:4:0/112
  Metric: 10         MT (IPv6 Unicast) IPv6 2001::99:4:4:0/112

 Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

Node5 is a Level-1-2 node. Each of the interfaces of Node5 is configured in a specific level. Interfaces Loopback0 and Gi0/0/0/1 are only in Level-2. Interfaces Loopback1 and Gi0/0/0/0 are only in Level-1.

Similar to Node3, Node5 is configured to propagate and leak host prefixes between levels. See the configuration of Node5 in Example 5-79.

Example 5-79: ISIS multi-area – Node5 ISIS configuration extract

```
interface Loopback0
 description Loopback
 ipv4 address 1.2.1.5 255.255.255.255
 ipv6 address 2001::1:2:1:5/128
!

RP/0/0/CPU0:xrvr-5#sh run int loop2
interface Loopback2
 ipv4 address 1.1.1.5 255.255.255.255
 ipv6 address 2001::1:1:1:5/128
!
route-policy LOOPBACKS
  if destination in (0.0.0.0/0 eq 32) or destination in (::/0
eq 128) then
    pass
  else
    drop
  endif
end-policy
!
router isis 1
 net 49.0002.0000.0000.0005.00
```

```
address-family ipv4 unicast
 metric-style wide
 propagate level 1 into level 2 route-policy LOOPBACKS
 propagate level 2 into level 1 route-policy LOOPBACKS
 segment-routing mpls
!
address-family ipv6 unicast
 metric-style wide
 propagate level 1 into level 2 route-policy LOOPBACKS
 propagate level 2 into level 1 route-policy LOOPBACKS
 segment-routing mpls
!
interface Loopback0
 passive
 circuit-type level-2-only
 address-family ipv4 unicast
  prefix-sid absolute 16205
 !
 address-family ipv6 unicast
  prefix-sid absolute 17205
 !
!
interface Loopback2
 passive
 circuit-type level-1
 address-family ipv4 unicast
  prefix-sid absolute 16105
 !
 address-family ipv6 unicast
  prefix-sid absolute 17105
 !
!
interface GigabitEthernet0/0/0/0
 circuit-type level-1
 point-to-point
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
!
interface GigabitEthernet0/0/0/1
 circuit-type level-2-only
 point-to-point
 address-family ipv4 unicast
```

```
  !
  address-family ipv6 unicast
  !
 !
```

Example 5-80 illustrates the ISIS L1 LSP originated by Node5 in Level-1. The Prefix-SIDs associated with the prefixes originated by Node1, 1.1.1.1/32 and 2001::1:1:1:1/128, still have the P-flag (PHP-off) and R-flag (Re-advertised) set. Node3 had already set these flags when it propagated these prefixes to Level-2.

Both P-flag and R-flag are also set for the propagated Prefix-SIDs of the prefixes that are originated by Node3 in Level-2, 1.2.1.3/32 and 2001::1:2:1:3/128. Node5 propagated these prefixes from Level-2 to Level-1.

Node5 also propagated the Level-2 prefixes local to Node5, 1.0.1.5/32 and 2001::1:0:1:5/128, to Level-1. Since these prefixes are local to Node5, the P-flag and R-flag are *not* set for the propagated Prefix-SIDs of these prefixes.

Notice that all prefixes that are propagated from Level-2 to Level-1 have their Up/Down bit set to mark that they have been propagated downwards, from L2 to L1. This bit is used to avoid loops by preventing propagating prefixes with Up/Down bit set. See IETF RFC 5302. Because the Up/Down-bit is set, they are displayed in the `show isis database` output with the `Interarea` suffix in the TLV name (.e.g. `IP-Extended-Interarea`).

Example 5-80: ISIS multi-area – L1 LSP advertised by Node5

```
RP/0/0/CPU0:xrvr-5#show isis database level 1 xrvr-5 verbose
```

```
IS-IS 1 (Level-1) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-5.00-00        * 0x00000021   0x4726
1191            1/0/0
  Area Address:   49.0002
  NLPID:          0xcc
  NLPID:          0x8e
  MT:             Standard (IPv4 Unicast)
  MT:             IPv6 Unicast
1/0/0
  Hostname:       xrvr-5
  IP Address:     1.2.1.5
  IPv6 Address:   2001::1:2:1:5
  Router Cap:     1.2.1.5, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 10         IS-Extended xrvr-6.00
    Interface IP Address: 99.5.6.5
    Neighbor IP Address: 99.5.6.6
    ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:30506
    ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:31506
  Metric: 40         IP-Extended-Interarea 1.1.1.1/32
    Prefix-SID Index: 101, Algorithm:0, R:1 N:1 P:1 E:0 V:0 L:0
  Metric: 30         IP-Extended-Interarea 1.1.1.2/32
    Prefix-SID Index: 102, Algorithm:0, R:1 N:1 P:1 E:0 V:0 L:0
  Metric: 20         IP-Extended-Interarea 1.1.1.3/32
    Prefix-SID Index: 103, Algorithm:0, R:1 N:1 P:1 E:0 V:0 L:0
  Metric: 0          IP-Extended 1.1.1.5/32
    Prefix-SID Index: 105, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 20         IP-Extended-Interarea 1.2.1.3/32
    Prefix-SID Index: 203, Algorithm:0, R:1 N:1 P:1 E:0 V:0 L:0
  Metric: 10         IP-Extended-Interarea 1.2.1.4/32
    Prefix-SID Index: 204, Algorithm:0, R:1 N:1 P:1 E:0 V:0 L:0
!! R-flag and P-flag are not set for prefix-SID of prefix
1.2.1.5/32
!! prefix 1.2.1.5/32 has been propagated from L2 to L1, but is
local to Node5
  Metric: 0          IP-Extended-Interarea 1.2.1.5/32
    Prefix-SID Index: 205, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 10         IP-Extended 99.5.6.0/24
  Metric: 10         MT (IPv6 Unicast) IS-Extended xrvr-6.00
    Interface IPv6 Address: 2001::99:5:6:5
    Neighbor IPv6 Address: 2001::99:5:6:6
    ADJ-SID: F:1 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:32506
    ADJ-SID: F:1 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:33506
  Metric: 40         MT (IPv6 Unicast) IPv6-Interarea
2001::1:1:1:1/128
    Prefix-SID Index: 1101, Algorithm:0, R:1 N:1 P:1 E:0 V:0
L:0
  Metric: 30         MT (IPv6 Unicast) IPv6-Interarea
2001::1:1:1:2/128
    Prefix-SID Index: 1102, Algorithm:0, R:1 N:1 P:1 E:0 V:0
L:0
  Metric: 20         MT (IPv6 Unicast) IPv6-Interarea
2001::1:1:1:3/128
    Prefix-SID Index: 1103, Algorithm:0, R:1 N:1 P:1 E:0 V:0
L:0
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:5/128
    Prefix-SID Index: 1105, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 20         MT (IPv6 Unicast) IPv6-Interarea
2001::1:2:1:3/128
    Prefix-SID Index: 1203, Algorithm:0, R:1 N:1 P:1 E:0 V:0
L:0
  Metric: 10         MT (IPv6 Unicast) IPv6-Interarea
2001::1:2:1:4/128
    Prefix-SID Index: 1204, Algorithm:0, R:1 N:1 P:1 E:0 V:0
L:0
!! R-flag and P-flag are not set for prefix-SID of prefix
2001::1:2:1:5/128
!! prefix 2001::1:2:1:5/128 has been propagated from L2 to L1,
but is local to Node5
  Metric: 0          MT (IPv6 Unicast) IPv6-Interarea
2001::1:2:1:5/128
    Prefix-SID Index: 1205, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
  Metric: 10         MT (IPv6 Unicast) IPv6 2001::99:5:6:0/112

 Total Level-1 LSP count: 1     Local Level-1 LSP count: 1
```

5.5.4 OSPFv2 Multi-Area example

Figure 5-35 shows the topology used for this OSPF inter-area example. It is a chain of seven nodes located in three areas. Node1, Node2, and Node3 are in area 1. Node3, Node4, and Node5 are in area 0. Node5, Node6, and Node7 are in area 2.

The ABR nodes have two loopback prefixes, one in each area.

All nodes allocated the same SRGB [16000-23999].

Prefix and Prefix-SID conventions:

- The Loopback IPv4 prefix of node N in area A: 1.A.1.N/32
- The IPv4 Prefix-SID of node N in area A is 16000 + 100*A + N.

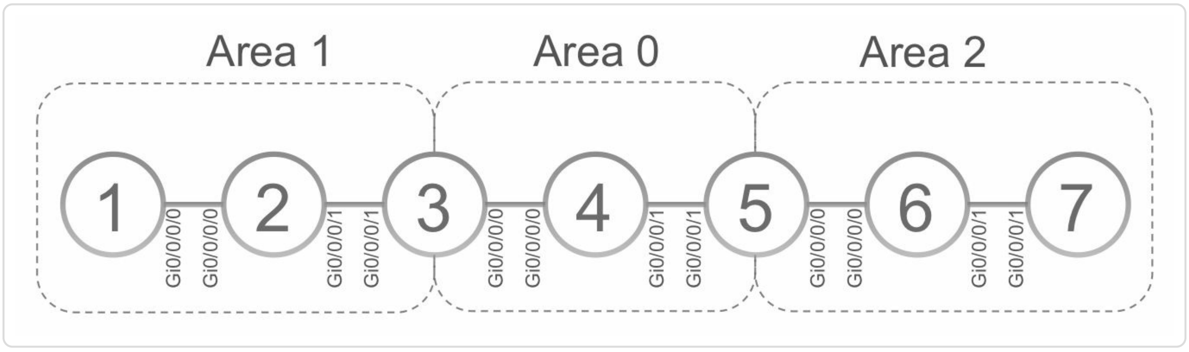


Figure 5-35: OSPF multi-area network topology

Figure 5-36 shows the loopback prefix and associated prefix-SID assignments for each node in each area. The column indicates the node and the row indicates the area. The cell content specifies the loopback prefix and its Prefix-SID as configured for that (node, area). For example Node3 has two loopbacks, one in Area1 (1.1.1.3/32, SID 16103) and one in Area0 (1.0.1.3/32, SID 16003).

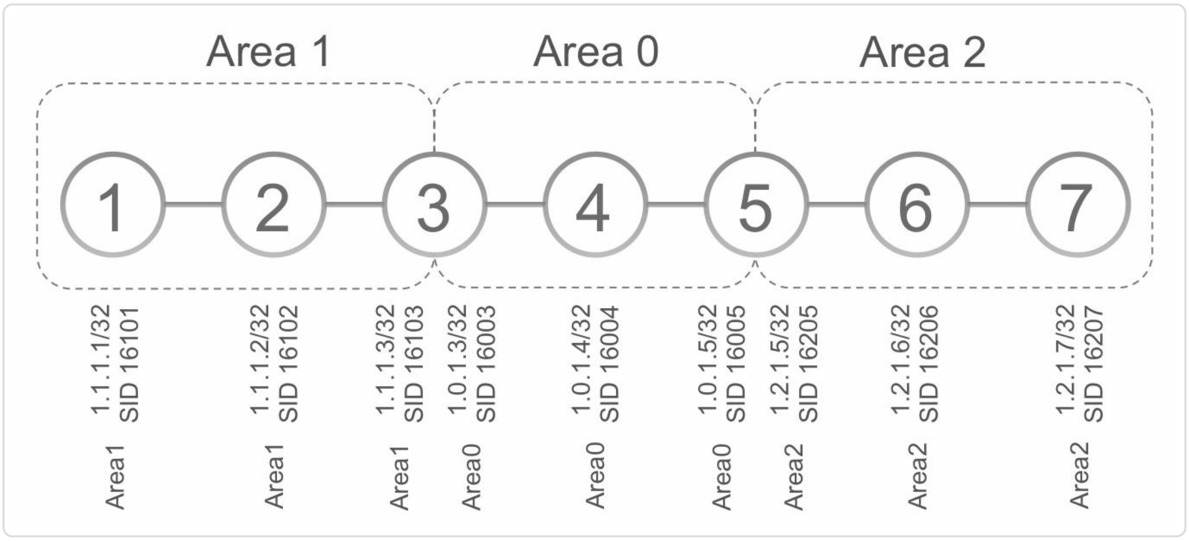


Figure 5-36: OSPF multi-area – IPv4 loopback prefixes and Prefix-SIDs

In the example the prefix advertisements are tracked from left to right. In particular the prefixes advertised by Node1, Node3 and Node5 are followed when they propagate through the areas to Area 2.

An IPv4 traceroute from Node7 to Node1 shows that there is end-to-end inter-area connectivity using the same Prefix-SID label 16101 throughout the whole path, until Node2 pops the label. See the traceroute output in Example 5-81.

Example 5-81: OSPF multi-area – end-to-end traceroute

```
RP/0/0/CPU0:xrvr-7#traceroute 1.1.1.1

Type escape sequence to abort.
Tracing the route to 1.1.1.1

 1  99.6.7.6 [MPLS: Label 16101 Exp 0 229 msec  29 msec  29
msec]
 2  99.5.6.5 [MPLS: Label 16101 Exp 0 19 msec  19 msec  29
msec]
 3  99.4.5.4 [MPLS: Label 16101 Exp 0 19 msec  29 msec  29
msec]
 4  99.3.4.3 [MPLS: Label 16101 Exp 0 29 msec  19 msec  19
msec]
```

```
 5  99.2.3.2 [MPLS: Label 16101 Exp 0 29 msec  29 msec  29
msec]
 6  99.1.2.1 29 msec  29 msec  29 msec
```

Node1 is located in OSPF area 1. Node1 advertises prefix 1.1.1.1/32 with Prefix-SID 16101. See the OSPF configuration of Node1 in Example 5-82.

Example 5-82: OSPF multi-area – OSPF configuration of Node1

```
interface Loopback0
 description Loopback in ospf area 1
 ipv4 address 1.1.1.1 255.255.255.255
!
router ospf 1
 router-id 1.1.1.1
 segment-routing mpls
area 1
  interface Loopback0
   passive enable
   prefix-sid absolute 16101
  !
  interface GigabitEthernet0/0/0/0
   cost 10
   network point-to-point
  !
!
```

Example 5-83 shows all the LSAs that Node1 originates in Area 1. The Router LSA (Type 1), the first LSA in the list with Link ID 1.1.1.1, contains the reachability information of Node1. The area-scoped (Type 10) Opaque LSAs at the bottom of the output are the Router Information LSA (Link ID 4.0.0.0), the Extended Prefix LSA (Link ID 7.0.0.1), and the Extended Link LSA (Link ID 8.0.0.3).

Example 5-83: OSPF multi-area – Node1 Area 1 LSAs

```
RP/0/0/CPU0:xrvr-1#show ospf database self-originate
```

```
            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum
Link count
1.1.1.1         1.1.1.1         1522        0x80000002 0x00948b
3

                Type-10 Opaque Link Area Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum
Opaque ID
4.0.0.0         1.1.1.1         1530        0x80000001
0x002c44        0
7.0.0.1         1.1.1.1         787         0x80000001
0x00181a        1
8.0.0.3         1.1.1.1         1522        0x80000002
0x00697a        3
```

Example 5-84 displays the Router LSA that Node1 advertises in Area 1. The loopback prefix 1.1.1.1/32 is advertised as a Stub network and the point-to-point adjacency with neighbor Node2 (router-id 1.1.1.2) is included as well. Also the network address of the interface is included as Stub network.

Example 5-84: OSPF multi-area – Node1 Area 1 Router LSA

```
RP/0/0/CPU0:xrvr-1#show ospf database router self-originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 1)

  LS age: 1630
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.1.1.1
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000002
```

```
    Checksum: 0x948b
    Length: 60
     Number of Links: 3

      Link connected to: a Stub Network
       (Link ID) Network/subnet number: 1.1.1.1
       (Link Data) Network Mask: 255.255.255.255
        Number of TOS metrics: 0
         TOS 0 Metrics: 1

      Link connected to: another Router (point-to-point)
       (Link ID) Neighboring Router ID: 1.1.1.2
       (Link Data) Router Interface address: 99.1.2.1
        Number of TOS metrics: 0
         TOS 0 Metrics: 10

      Link connected to: a Stub Network
       (Link ID) Network/subnet number: 99.1.2.0
       (Link Data) Network Mask: 255.255.255.0
        Number of TOS metrics: 0
         TOS 0 Metrics: 10
```

Example 5-85 displays the Router Information Opaque LSA (Opaque-type 4) advertised by Node1 in Area 1. This LSA has a fixed Link ID (4.0.0.0) and contains a number of TLVs. The first TLV is the Router Information Capabilities TLV that indicates the OSPF capabilities of this node as defined in IETF RFC 7770 and its later extensions. The next TLV is the Segment Routing Algorithm TLV that indicates which prefix reachability computation algorithms are supported by this node. In this example Node1 supports the well-known IGP metric SPF Algorithm (type 0) and the Strict SPF Algorithm (type 1). The last TLV in this example is the Segment Routing Range TLV that contains the SRGB settings of Node1: Node1 allocated SRGB label range [16000-23999] in this example.

Example 5-85: OSPF multi-area – Node1 Area 1 Router Information LSA

```
RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 4.0.0.0 self-
```

```
originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 1)

  LS age: 1709
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 4.0.0.0
  Opaque Type: 4
  Opaque ID: 0
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000001
  Checksum: 0x2c44
  Length: 52

    Router Information TLV: Length: 4
    Capabilities:
      Graceful Restart Helper Capable
      Stub Router Capable
      All capability bits: 0x60000000

    Segment Routing Algorithm TLV: Length: 2
      Algorithm: 0
      Algorithm: 1

    Segment Routing Range TLV: Length: 12
      Range Size: 8000

        SID sub-TLV: Length 3
         Label: 16000
```

Example 5-86 shows the Extended Prefix Opaque LSA (Opaque-type 7) advertised by Node1 in Area 1. This LSA contains the Extended Prefix TLV that specifies the prefix that this TLV is associated with. In this example it is the Intra-Area prefix 1.1.1.1/32. Since this prefix has the Node-flag set (N:1) it identifies the originating node. The Extended Prefix

TLV contains one sub-TLV: the Prefix-SID sub-TLV (`SID sub-TLV` in the output). This sub-TLV contains the Prefix-SID that is associated with prefix 1.1.1.1/32: SID index 101. With an SRGB Base value of 16000, the label value for this Prefix-SID is 16101 (=16000 + 101). None of the Prefix-SID flags are set; therefore Node1 requests the default PHP behavior from the penultimate hop of this Prefix Segment (NP-flag=0). The algorithm used for this Prefix-SID is the default SPF (type 0).

Example 5-86: OSPF multi-area – Node1 Area 1 Extended Prefix LSA

```
RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 1.1.1.1/32
self-originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 1)

  LS age: 1364
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.1
  Opaque Type: 7
  Opaque ID: 1
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000001
  Checksum: 0x181a
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 1              !! Intra-area prefix
      AF        : 0
      Flags     : 0x40           !! A:0; N:1
      Prefix    : 1.1.1.1/32

      SID sub-TLV: Length: 8
        Flags     : 0x0          !! (NA); NP:0; M:0; E:0; V:0;
L:0
        MTID      : 0
        Algo      : 0            !! SPF
        SID Index : 101
```

Example 5-87 displays the Extended Link Opaque LSA (Opaque-type 8) advertised by Node1 in Area 1. This LSA contains the link attributes for the adjacency of Node1 to Node2. Adjacencies themselves are not propagated between areas, therefore the Extended Link LSA, which specifies additional attributes of the adjacency it is associated with, is not propagated between areas either. The Opaque ID of this LSA (3 in this example) is only used to differentiate between multiple Opaque LSAs of the same Opaque Type advertised by the same node. The Extended Link TLV contains the sub-TLVs for the Adjacency-SIDs (one protection-eligible, B:1, and one unprotected, B:0) and the sub-TLV specifying the neighbor's interface address of the link (99.1.2.2 in this example).

Example 5-87: OSPF multi-area – Node1 Area 1 Extended Link LSA

```
RP/0/0/CPU0:xrvr-1#show ospf database opaque-area 8.0.0.3 self-
originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 1)

  LS age: 483
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 8.0.0.3
  Opaque Type: 8
  Opaque ID: 3
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000003
  Checksum: 0x677b
  Length: 68

    Extended Link TLV: Length: 44
      Link-type : 1             !! point-to-point
      Link ID   : 1.1.1.2       !! Neighboring Router ID
```

```
     Link Data : 99.1.2.1        !! Router Interface address

    Adj sub-TLV: Length: 7
       Flags     : 0xe0          !! B:1,V:1,L:1,S:0
       MTID      : 0
       Weight    : 0
       Label     : 30102

    Adj sub-TLV: Length: 7
       Flags     : 0x60          !! B:0,V:1,L:1,S:0
       MTID      : 0
       Weight    : 0
       Label     : 31102

    Remote If Address sub-TLV: Length: 4
       Neighbor Address: 99.1.2.2
```

The LSAs described above where advertised by Node1 in OSPF Area 1. The Area Border Router (ABR) Node3 is connected to Area 1 and Area 0 and propagates these prefixes from Area 1 to Area 0. Example 5-89 displays the configuration of ABR Node3. Node3 has two loopback interfaces, one in each area: Loopback0 in Area 0 and Loopback1 in Area 1.

Example 5-88: OSPF multi-area – OSPF configuration of Node3

```
interface Loopback0
 ipv4 address 1.0.1.3 255.255.255.255
!
interface Loopback1
 ipv4 address 1.1.1.3 255.255.255.255
!
router ospf 1
 router-id 1.1.1.3
 segment-routing mpls
 area 0
  interface Loopback0
   passive enable
   prefix-sid absolute 16003
  !
```

```
   interface GigabitEthernet0/0/0/0
    cost 10
    network point-to-point
   !
  !
  area 1
   interface Loopback1
    passive enable
    prefix-sid absolute 16103
   !
   interface GigabitEthernet0/0/0/1
    cost 10
    network point-to-point
   !
  !
 !
```

Example 5-89 displays all the LSAs that Node3 originates in Area 0. Node3 advertises it own reachability information in its Router LSA and the reachability information of the propagated prefixes in the Summary LSAs. It also advertises the Segment Routing information of both intra- and inter-area prefixes in a number of Area-scope (Type 10) Opaque-LSAs. Note: to restrict the display to the information of an OSPF instance "I" and Area "A", use the command `show ospf I A (…)`.

Example 5-89: OSPF multi-area – Node3 Area 0 LSAs

```
RP/0/0/CPU0:xrvr-3#show ospf 1 0 database self-originate


            OSPF Router with ID (1.1.1.3) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
Link count
1.1.1.3         1.1.1.3         466         0x80000005 0x00c245
3

                Summary Net Link States (Area 0)
```

```
Link ID         ADV Router      Age         Seq#       Checksum
1.1.1.1         1.1.1.3         1466        0x80000002 0x00f922
1.1.1.2         1.1.1.3         1466        0x80000002 0x008b99
1.1.1.3         1.1.1.3         1466        0x80000002 0x001d11
99.1.2.0        1.1.1.3         1466        0x80000002 0x00efca
99.2.3.0        1.1.1.3         1466        0x80000002 0x00744e
99.2.10.0       1.1.1.3         1466        0x80000002 0x008b26
99.2.11.0       1.1.1.3         1466        0x80000002 0x008030

                Type-10 Opaque Link Area Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
Opaque ID
4.0.0.0         1.1.1.3         1466        0x80000002
0x001656        0
7.0.0.2         1.1.1.3         728         0x80000003
0x0054d1        2
7.0.0.3         1.1.1.3         1466        0x80000002
0x00e8fe        3
7.0.0.4         1.1.1.3         728         0x80000002
0x00b632        4
7.0.0.5         1.1.1.3         466         0x80000002
0x00eb9f        5
8.0.0.4         1.1.1.3         1466        0x80000003
0x00a41f        4
```

Example 5-90 displays the Router LSA (type 1 LSA) as advertised by Node3 in Area 0. This LSA contains the Node3 loopback prefix 1.0.1.3/32, which is located in Area 0.

Example 5-90: OSPF multi-area – Node3 Area 0 Router LSA

```
RP/0/0/CPU0:xrvr-3#show ospf 1 0 database router self-originate


            OSPF Router with ID (1.1.1.3) (Process ID 1)

                Router Link States (Area 0)

  LS age: 582
  Options: (No TOS-capability, DC)
```

```
    LS Type: Router Links
    Link State ID: 1.1.1.3
    Advertising Router: 1.1.1.3
    LS Seq Number: 80000005
    Checksum: 0xc245
    Length: 60
    Area Border Router
     Number of Links: 3

      Link connected to: a Stub Network
       (Link ID) Network/subnet number: 1.0.1.3
       (Link Data) Network Mask: 255.255.255.255
        Number of TOS metrics: 0
         TOS 0 Metrics: 1

      Link connected to: another Router (point-to-point)
       (Link ID) Neighboring Router ID: 1.1.1.4
       (Link Data) Router Interface address: 99.3.4.3
        Number of TOS metrics: 0
         TOS 0 Metrics: 10

      Link connected to: a Stub Network
       (Link ID) Network/subnet number: 99.3.4.0
       (Link Data) Network Mask: 255.255.255.0
        Number of TOS metrics: 0
         TOS 0 Metrics: 10
```

Node3 also advertises the Extended Prefix LSA associated with prefix 1.0.1.3/32 in area 0. This LSA advertises the Prefix-SID of 1.0.1.3/32, SID index 3. See the output in Example 5-91. 1.0.1.3/32 is a node prefix since the N-flag is set. None of the Prefix-SID flags are set, so Node3 requests the default PHP behavior from the penultimate hop.

Example 5-91: OSPF multi-area – Node3 Area 0 Extended Prefix LSA 1.0.1.3/32

```
RP/0/0/CPU0:xrvr-3#show ospf 1 0 database opaque-area
1.0.1.3/32 self-originate


            OSPF Router with ID (1.1.1.3) (Process ID 1)
```

```
                    Type-10 Opaque Link Area Link States (Area 0)

  LS age: 706
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.5
  Opaque Type: 7
  Opaque ID: 5
  Advertising Router: 1.1.1.3
  LS Seq Number: 80000002
  Checksum: 0xeb9f
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 1                !! Intra-area prefix
      AF        : 0
      Flags     : 0x40             !! A:0; N:1
      Prefix    : 1.0.1.3/32

      SID sub-TLV: Length: 8
        Flags     : 0x0            !! (NA); NP:0; M:0; E:0; V:0;
L:0
        MTID      : 0
        Algo      : 0              !! SPF
        SID Index : 3
```

Node3 also has a loopback prefix configured in area 1: 1.1.1.3/32. Node3 propagates this prefix from area 1 to area 0 and advertises it as an inter-area prefix in one of its Area 0 Summary LSAs. The output in Example 5-92 shows this Summary LSA of 1.1.1.3/32.

Example 5-92: OSPF multi-area – Node3 Area 0 Summary LSA 1.1.1.3/32

```
RP/0/0/CPU0:xrvr-3#show ospf 1 0 database summary 1.1.1.3 self-
originate


            OSPF Router with ID (1.1.1.3) (Process ID 1)
```

```
                    Summary Net Link States (Area 0)

  LS age: 1913
  Options: (No TOS-capability, DC)
  LS Type: Summary Links (Network)
  Link State ID: 1.1.1.3 (Summary Network Number)
  Advertising Router: 1.1.1.3
  LS Seq Number: 80000002
  Checksum: 0x1d11
  Length: 28
  Network Mask: /32
        TOS: 0  Metric: 1
```

Node3 also propagates the associated Prefix-SID of 1.1.1.3/32 from Area 1 to Area 0. Node3 therefore originates an Extended Prefix LSA associated with prefix 1.1.1.3/32 in Area 0. This LSA advertises the Prefix-SID of 1.1.1.3/32, SID index 103. See the output of this LSA in Example 5-93. Even though the prefix 1.1.1.3/32 has been propagated between areas and is advertised as an inter-area prefix in area 0, the Prefix-SID flags have not been updated. The flags still indicate regular PHP behavior (NP:0). The reason is that this prefix is local to the ABR Node3, therefore the penultimate hop should pop the label before forwarding to ABR Node3.

At the time of writing this book, Cisco IOS-XR OSPF implementation did not support the A-flag (Attached) in the Extended Prefix TLV. An ABR should set this flag when it originates a local or attached prefix as an inter-area prefix in a connected area. The A-flag would be set by Node3 in the Extended Prefix TLV for the inter-area prefix 1.1.1.3/32 that Node3 originates in Area 0, since 1.1.1.3/32 is local to Node3. ABR Node5 would clear the A-flag again when it propagates the prefix 1.1.1.3/32 from Area 0 to Area 2, since this prefix is not local or attached to ABR Node5.

Example 5-93: OSPF multi-area – Node3 Area 0 Extended Prefix LSA 1.1.1.3/32

```
RP/0/0/CPU0:xrvr-3#show ospf 1 0 database opaque-area
1.1.1.3/32 self-originate


            OSPF Router with ID (1.1.1.3) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 0)

  LS age: 1276
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.2
  Opaque Type: 7
  Opaque ID: 2
  Advertising Router: 1.1.1.3
  LS Seq Number: 80000003
  Checksum: 0x54d1
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 3                 !! Inter-area prefix
      AF        : 0
      Flags     : 0x40              !! A:0; N:1
      Prefix    : 1.1.1.3/32

      SID sub-TLV: Length: 8
        Flags     : 0x0             !! (NA); NP:0; M:0; E:0; V:0;
L:0
        MTID      : 0
        Algo      : 0               !! SPF
        SID Index : 103
```

Node3 also propagates the loopback prefix of Node1, 1.1.1.1/32, from Area 1 to Area 0. Therefore Node3 originates a Summary LSA for this prefix and the associated Extended Prefix LSA. Example 5-94 shows the Summary LSA for prefix 1.1.1.1/32 in Area 0.

Example 5-94: OSPF multi-area – Node3 Area 0 Summary LSA 1.1.1.1/32

```
RP/0/0/CPU0:xrvr-3#show ospf 1 0 database summary 1.1.1.1 self-
```

```
originate


            OSPF Router with ID (1.1.1.3) (Process ID 1)

                Summary Net Link States (Area 0)

  LS age: 161
  Options: (No TOS-capability, DC)
  LS Type: Summary Links (Network)
  Link State ID: 1.1.1.1 (Summary Network Number)
  Advertising Router: 1.1.1.3
  LS Seq Number: 80000003
  Checksum: 0xf723
  Length: 28
  Network Mask: /32
        TOS: 0  Metric: 21
```

The Extended Prefix LSA advertised by Node3 in Area 0 for inter-area prefix 1.1.1.1/32, contains the Prefix-SID index 101 associated with this prefix. See the output of this LSA in Example 5-95. The NP-flag (PHP-off) has been set for this Prefix-SID. This means that the penultimate hop must not pop the label of packets on this Prefix Segment before forwarding to ABR Node3. This is indeed the correct behavior since prefix 1.1.1.1/32 is not local to Node3 and a packet on the Prefix Segment to Node1 needs to keep its Segment Routing transport label until it reaches the real penultimate hop of this Prefix Segment (Node2 in this example). This way a continuous label switched path towards Node1 is achieved.

Example 5-95: OSPF multi-area – Node3 Area 0 Extended Prefix LSA 1.1.1.1/32

```
RP/0/0/CPU0:xrvr-3#show ospf 1 0 database opaque-area
1.1.1.1/32 self-originate


            OSPF Router with ID (1.1.1.3) (Process ID 1)
```

```
                Type-10 Opaque Link Area Link States (Area 0)

  LS age: 1484
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.4
  Opaque Type: 7
  Opaque ID: 4
  Advertising Router: 1.1.1.3
  LS Seq Number: 80000002
  Checksum: 0xb632
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 3                 !! Inter-area prefix
      AF        : 0
      Flags     : 0x40              !! A:0; N:1
      Prefix    : 1.1.1.1/32

      SID sub-TLV: Length: 8
        Flags     : 0x40            !! (NA); NP:1; M:0; E:0; V:0;
L:0
        MTID      : 0
        Algo      : 0               !! SPF
        SID Index : 101
```

Node5 is an ABR connected to Area 0 and Area 2. The configuration of Node5 is shown in Example 5-96. ABR Node5 has two loopbacks, one in Area 0 (Loopback0) and one in Area 2 (Loopback2).

Example 5-96: OSPF multi-area – OSPF configuration of Node5

```
interface Loopback0
 ipv4 address 1.0.1.5 255.255.255.255
!
interface Loopback2
 ipv4 address 1.2.1.5 255.255.255.255
!
router ospf 1
 router-id 1.1.1.5
 segment-routing mpls
```

```
  area 0
   interface Loopback0
    passive enable
    prefix-sid absolute 16005
   !
   interface GigabitEthernet0/0/0/1
    cost 10
    network point-to-point
   !
  !
  area 2
   interface Loopback2
    passive enable
    prefix-sid absolute 16205
   !
   interface GigabitEthernet0/0/0/0
    cost 10
    network point-to-point
   !
  !
 !
```

Example 5-97 displays the LSAs that ABR Node5 originates in Area 2. Node5 originates its own reachability information in its Router LSA and the reachability information of the propagated prefixes in the Summary LSAs. It also advertises the SID information in a number of Opaque-LSAs. The advertisement of Node5's local prefixes is equivalent to the other ABR Node3 and is not covered here.

Example 5-97: OSPF multi-area – Node5 Area 2 LSAs

```
RP/0/0/CPU0:xrvr-5#show ospf 1 2 database self-originate


            OSPF Router with ID (1.1.1.5) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
Link count
```

```
1.1.1.5         1.1.1.5         1568        0x80000005 0x00f7fc
3

                Summary Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
1.0.1.3         1.1.1.5         1568        0x80000002 0x00e533
1.0.1.4         1.1.1.5         1568        0x80000002 0x0077aa
1.0.1.5         1.1.1.5         1568        0x80000002 0x000922
1.1.1.1         1.1.1.5         555         0x80000003 0x00b450
1.1.1.2         1.1.1.5         555         0x80000003 0x0046c7
1.1.1.3         1.1.1.5         555         0x80000003 0x00d73f
99.1.2.0        1.1.1.5         555         0x80000003 0x00aaf8
99.2.3.0        1.1.1.5         555         0x80000003 0x002f7c
99.2.10.0       1.1.1.5         555         0x80000003 0x004654
99.2.11.0       1.1.1.5         555         0x80000003 0x003b5e
99.3.4.0        1.1.1.5         555         0x80000003 0x00b3ff
99.4.5.0        1.1.1.5         555         0x80000003 0x003883
99.4.8.0        1.1.1.5         555         0x80000003 0x007b33
99.4.9.0        1.1.1.5         555         0x80000003 0x00703d
99.4.10.0       1.1.1.5         555         0x80000003 0x006547

                Type-10 Opaque Link Area Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
Opaque ID
4.0.0.0         1.1.1.5         555         0x80000003
0x000861        0
7.0.0.2         1.1.1.5         1568        0x80000003
0x00c8f3        2
7.0.0.3         1.1.1.5         1568        0x80000002
0x005233        3
7.0.0.4         1.1.1.5         1568        0x80000002
0x00e461        4
7.0.0.5         1.1.1.5         1816        0x80000003
0x00eef1        5
7.0.0.6         1.1.1.5         555         0x80000003
0x00bc25        6
7.0.0.7         1.1.1.5         1816        0x80000002
0x008c57        7
7.0.0.8         1.1.1.5         1568        0x80000002
0x0094af        8
8.0.0.4         1.1.1.5         555         0x80000004
0x00c0ee        4
```

Besides originating LSAs for its own prefixes, Node5 also propagates the loopback prefixes of the other nodes as inter-area prefixes, with their associated Extended Prefix LSA. Node5 propagates for example the loopback prefix of Node1, 1.1.1.1/32, from Area 0 to Area 2.. Example 5-98 shows the Summary LSA for prefix 1.1.1.1/32 as advertised by Node5 in Area 2.

Example 5-98: OSPF multi-area – Node5 Area 2 Summary LSA 1.1.1.1/32

```
RP/0/0/CPU0:xrvr-5#show ospf 1 2 database summary 1.1.1.1 self-
originate


            OSPF Router with ID (1.1.1.5) (Process ID 1)

                Summary Net Link States (Area 2)

  LS age: 1797
  Options: (No TOS-capability, DC)
  LS Type: Summary Links (Network)
  Link State ID: 1.1.1.1 (Summary Network Number)
  Advertising Router: 1.1.1.5
  LS Seq Number: 8000001a
  Checksum: 0x8667
  Length: 28
  Network Mask: /32
        TOS: 0  Metric: 41
```

Node5 also propagates the Prefix-SID of 1.1.1.1/32 to Area 2. Example 5-99 shows the Extended Prefix LSA for prefix 1.1.1.1/32 as advertised by Node5 in Area 2. The NP-flag (PHP-off) is set for the Prefix-SID; the upstream neighbor (Node6 in this example) must not pop the label for packets on this Prefix Segment before forwarding to the ABR Node5.

Example 5-99: OSPF multi-area – Node5 Area 2 Extended Prefix LSA

```
RP/0/0/CPU0:xrvr-5#show ospf 1 2 database opaque-area
1.1.1.1/32 self-originate


            OSPF Router with ID (1.1.1.5) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 2)

  LS age: 1101
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.7
  Opaque Type: 7
  Opaque ID: 7
  Advertising Router: 1.1.1.5
  LS Seq Number: 8000001a
  Checksum: 0x5c6f
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 3              !! Inter-area prefix
      AF        : 0
      Flags     : 0x40           !! A:0; N:1
      Prefix    : 1.1.1.1/32

      SID sub-TLV: Length: 8
        Flags     : 0x40         !! (NA); NP:1; M:0; E:0; V:0;
L:0
        MTID      : 0
        Algo      : 0            !! SPF
        SID Index : 101
```

5.6 Redistribution with Prefix-SIDs

Prefixes with Prefix-SIDs can be redistributed between IGP and BGP domains. The Prefix-SID follows its prefix and is advertised with the prefix in the other SR Domain.

The protocol that takes prefixes from another protocol and advertises them is called the *redistributing protocol*. The protocol from which the prefixes are redistributed is called the *redistributed protocol*.

REMINDER: Redistribution

When redistributing prefixes between routing protocols, they are not redistributed directly from one protocol to the other. The routing protocol that redistributes the prefixes from another protocol, only takes the routes that the redistributed protocol installed in RIB and the local prefixes that the redistributed protocol advertises. RIB is the intermediate step in the redistribution process. The redistributing protocol does not (re-)install the redistributed prefixes into RIB.

When a protocol redistributes a prefix, the protocol uses the information that RIB provides. RIB passes the local label of the prefix with the prefix. In the case of SR, that local label is the Prefix-SID of the prefix. The redistributing protocol then advertises the prefix with its Prefix-SID (as its local label). Only valid Prefix-SIDs are redistributed, those that are installed in RIB.

To redistribute Prefix-SIDs between protocols, it is required that both protocols use the same SRGB. If not, then the prefixes are redistributed without their Prefix-SIDs.

The redistributing routing protocol makes no distinction between a Prefix-SID that is natively advertised with a prefix and a Prefix-SID that is advertised by a Mapping Server (described further in chapter 8, "Segment Routing Mapping Server"). If the redistributed protocol installs the Prefix-SID in RIB then it can be redistributed, regardless if the Prefix-SID is native or SRMS advertised. Note that the redistributing protocol advertises the prefix-SIDs as *native* Prefix-SIDs, even if they are SRMS advertised Prefix-SIDs in the redistributed protocol.

Redistribution of local prefixes does not redistribute their Prefix-SIDs. To advertise a local prefix of the border node with its Prefix-SID in the different domains, the loopback interface of that prefix must be configured under the different protocols.

Redistribution is important in many existing network designs. The propagation of Prefix-SIDs from the source to the destination protocol/instance ensures that Prefix Segments operate seamlessly along with their prefixes. From designs where the same or different IGP protocol instances are used to partition networks to architectures like the Inter-AS Option C and Seamless MPLS, Segment Routing allows for migration of existing networks. We will also look at new solutions in chapter 10, "Large Scale Interconnect with Segment Routing" that allow for interconnection of large scale networks.

Further in this section, we shall look closer at how redistribution works and in doing so we will also look into the BGP Control Plane operations for Segment Routing. One can first go through BGP Segment Routing operations covered in chapter 6, "Segment Routing BGP Control Plane" before coming back to this section and continuing further.

HIGHLIGHT

Route redistribution between IGP protocols or instances and between IGP and BGP works seamlessly along with Segment Routing.

It is required that IGP/BGP instances use the same SRGB when doing redistribution.

Prefix-SIDs learnt from the source protocol/instance are propagated into the destination protocol/instance as Prefix-SIDs; Prefix-SIDs were designed to be Global Segment across the SR Domain.

5.6.1 Redistribution Examples

The network topology in Figure 5-37 is used to illustrate the redistribution behavior. The network consists of three SR Domains: an SR ISIS domain (Node1 and Node2), an SR OSPF domain (Node2 and Node3), and an SR BGP domain (Node3 and Node4). Each node *N* advertises a prefix 1.1.1.*N*/32 with Prefix-SID 16000 + *N*. The border nodes Node2 and Node3 mutually redistribute all host prefixes with their Prefix-SIDs between the domains.

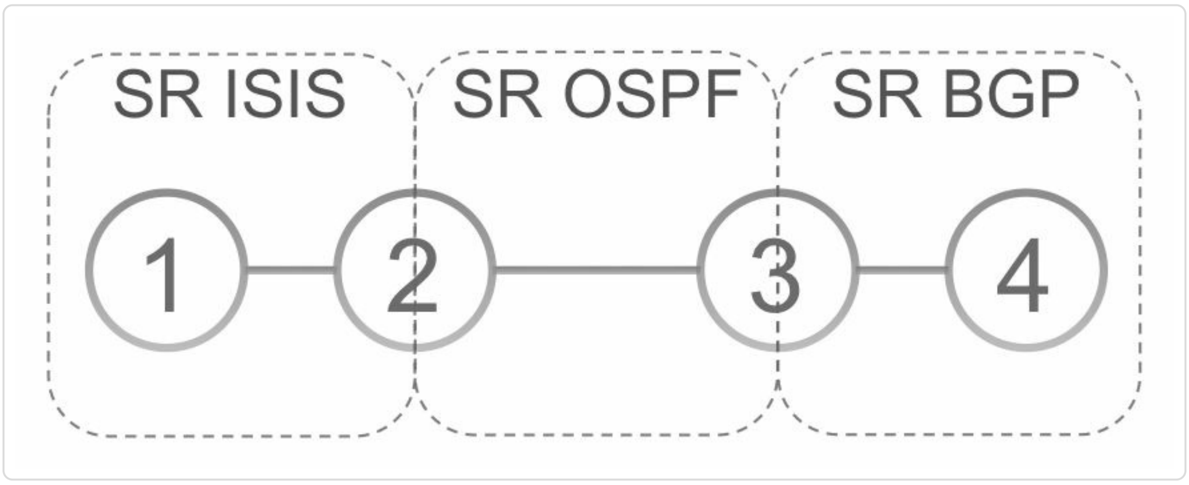


Figure 5-37: Network topology illustrating redistribution

5.6.1.1 Redistributing between IGPs

This example discusses the redistribution between ISIS and OSPF. Node1 originates prefix 1.1.1.1/32 with its Prefix-SID 16001. The border node Node2 installs this prefix in RIB, as displayed in Example 5-100. This RIB entry is installed by `isis 1` (line 4). The local label is the Prefix-SID label 16001 (line 16). The outgoing label 0x100004 (1048580) on line 9 is an internal label value that represents “pop”; Node2 will pop the label of packets on this Prefix Segment since it is the penultimate hop for this Prefix Segment.

Example 5-100: RIB entry of Node1’s ISIS prefix on Node2

```
 1 RP/0/0/CPU0:xrvr-2#show route 1.1.1.1/32 detail
 2
 3 Routing entry for 1.1.1.1/32
 4   Known via "isis 1", distance 115, metric 20, labeled SR,
type level-2
 5   Installed Aug  1 11:26:36.309 for 00:30:26
 6   Routing Descriptor Blocks
 7     99.1.2.1, from 1.1.1.1, via GigabitEthernet0/0/0/0
 8       Route metric is 20
 9       Label: 0x100004 (1048580)
10       Tunnel ID: None
11       Binding Label: None
12       Extended communities count: 0
13       Path id:1        Path ref count:0
14       NHID:0x1(Ref:2)
15   Route version is 0x2 (2)
16   Local Label: 0x3e81 (16001)
17   IP Precedence: Not Set
18   QoS Group ID: Not Set
19   Flow-tag: Not Set
20   Fwd-class: Not Set
21   Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD
Type RIB_SVD_TYPE_LOCAL
22   Download Priority 1, Download Version 8
23   No advertising protos.
```

OSPF on Node2 is configured to redistribute all host prefixes from ISIS instance 1, as shown on line 12 of the OSPF configuration in Example 5-101. Redistribution is limited with the route-policy `LOOPBACKS`, which only permits host prefixes (/32).

Example 5-101: OSPF configuration on Node2 to redistribute ISIS into OSPF

```
 1 route-policy LOOPBACKS
 2   if destination in (0.0.0.0/0 eq 32) then
 3     pass
 4   else
 5     drop
 6   endif
 7 end-policy
 8 !
 9 router ospf 2
10  segment-routing mpls
11  redistribute isis 1 route-policy LOOPBACKS
12  area 0
13   interface Loopback0
14    passive enable
15    prefix-sid absolute 16002
16   !
17   interface GigabitEthernet0/0/0/1
18    network point-to-point
19   !
20  !
21 !
```

Example 5-102 shows the list of OSPF LSAs that Node2 originates. Node2 advertises the Router LSA (line 9) and the Type-10 LSAs (line 14-16) for its local OSPF prefixes. Of interest in this section are the External LSA for prefix 1.1.1.1/32 on line 21 and the Opaque LSA associated with that prefix on line 26. Notice that the Opaque LSA is a `Type-11` LSA (line 23). A type 11 LSA has Autonomous System (AS) flooding scope; it is an Opaque LSA that is flooded throughout the AS.

Example 5-102: OSPF LSAs originated by Node2

```
RP/0/0/CPU0:xrvr-2#show ospf database self-originate


            OSPF Router with ID (1.1.1.2) (Process ID 2)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.1.1.2         1.1.1.2         320         0x80000003 0x00aa7b 3

                Type-10 Opaque Link Area Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Opaque ID
4.0.0.0         1.1.1.2         320         0x80000002 0x00244a        0
7.0.0.1         1.1.1.2         320         0x80000002 0x00068c        1
8.0.0.5         1.1.1.2         320         0x80000003 0x0022b6        5

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
1.1.1.1         1.1.1.2         320         0x80000002 0x009303 0

                Type-11 Opaque Link AS Link States

Link ID         ADV Router      Age         Seq#       Checksum Opaque ID
7.0.0.1         1.1.1.2         320         0x80000002 0x002e61        1
```

Example 5-103 shows the content of the External LSA that Node2 originates for prefix 1.1.1.1/32.

Example 5-103: External LSA originated by Node2

```
RP/0/0/CPU0:xrvr-2#show ospf database external self-originate


            OSPF Router with ID (1.1.1.2) (Process ID 2)

                Type-5 AS External Link States

  LS age: 440
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 1.1.1.1 (External Network Number)
  Advertising Router: 1.1.1.2
  LS Seq Number: 80000002
  Checksum: 0x9303
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

Example 5-104 displays the Extended Prefix LSA that Node2 originates for prefix 1.1.1.1/32. The Extended Prefix TLV identifies the prefix (lines 19 to 23); it is an external prefix (`Route-type: 5`) that is not a node prefix (N:0) and that is not attached to the ABR (A:0). The Prefix-SID sub-TLV specifies the Prefix-SID and its attributes (lines 25 to 29). The SID index is 1 and the PHP-off flag is set (NP:1).

Example 5-104: Extended Prefix LSA originated by Node2

```
 1 RP/0/0/CPU0:xrvr-2#show ospf database opaque-as self-
originate
 2
 3
 4             OSPF Router with ID (1.1.1.2) (Process ID 2)
 5
 6                 Type-11 Opaque Link AS Link States
```

```

  LS age: 509
  Options: (No TOS-capability, DC)
  LS Type: Opaque AS Link
  Link State ID: 7.0.0.1
  Opaque Type: 7
  Opaque ID: 1
  Advertising Router: 1.1.1.2
  LS Seq Number: 80000002
  Checksum: 0x2e61
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 5              !! External prefix
      AF        : 0
      Flags     : 0x0            !! A:0; N:0
      Prefix    : 1.1.1.1/32

      SID sub-TLV: Length: 8
        Flags     : 0x40         !! (NA); NP:1; M:0; E:0; V:0; L:0
        MTID      : 0
        Algo      : 0            !! SPF
        SID Index : 1
```

Node2 also redistributes in the other direction; it redistributes the OSPF host prefixes into ISIS. Node2 for example redistributes Node3's loopback prefix 1.1.1.3/32 with its Prefix-SID 16003 into the ISIS domain. Node2's RIB entry for prefix 1.1.1.3/32 is displayed in Example 5-105. The RIB entry is installed by `ospf 2` (line 4). The local label value for this prefix is the Prefix-SID label 16003 (line 17). The outgoing label is 3 (implicit-null) since Node2 is the penultimate hop for the Prefix Segment of Node3.

Example 5-105: RIB entry of Node3's OSPF prefix on Node2

```
1 RP/0/0/CPU0:xrvr-2#show route 1.1.1.3/32 detail
2
3 Routing entry for 1.1.1.3/32
4   Known via "ospf 2", distance 110, metric 2, labeled SR, type intra area
```

```
  Installed Aug  1 11:26:40.949 for 00:57:37
  Routing Descriptor Blocks
    99.2.3.3, from 1.1.1.3, via GigabitEthernet0/0/0/1
      Route metric is 2
      Label: 0x3 (3)
      Tunnel ID: None
      Binding Label: None
      Extended communities count: 0
      Path id:1       Path ref count:0
      NHID:0x2(Ref:5)
      OSPF area: 0
  Route version is 0x2 (2)
  Local Label: 0x3e83 (16003)
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: Not Set
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD Type RIB_SVD_TYPE_LOCAL
  Download Priority 1, Download Version 11
  No advertising protos.
```

ISIS is configured to redistribute the OSPF host prefixes, as indicated in line 14 of the ISIS configuration in Example 5-106. The route-policy `LOOPBACKS` only permits /32 prefixes.

Example 5-106: ISIS configuration on Node2 to redistribute OSPF into ISIS

```
 1 route-policy LOOPBACKS
 2   if destination in (0.0.0.0/0 eq 32) then
 3     pass
 4   else
 5     drop
 6   endif
 7 end-policy
 8 !
 9 router isis 1
10  is-type level-2-only
11  net 49.0001.0000.0000.0002.00
12  address-family ipv4 unicast
13   metric-style wide
14   redistribute ospf 2 route-policy LOOPBACKS
```

```
  segment-routing mpls
 !
 interface Loopback0
  address-family ipv4 unicast
   prefix-sid absolute 16002
  !
 !
 interface GigabitEthernet0/0/0/0
  point-to-point
  address-family ipv4 unicast
  !
 !
!
```

Example 5-107 shows the ISIS LSP that Node2 originates. The IP Reachability TLV for prefix 1.1.1.3/32 is displayed in lines 19 and 20. The Prefix-SID index is 3. The Prefix-SID flags R (Re-advertisement) and P (PHP-off) are set (`R:1` and `P:1` in line 20), which means that the prefix is re-advertised (redistributed) and no Penultimate Hop Popping (PHP) must be done for this Prefix-SID. PHP-off flag is set since Node2 is not the ultimate hop for the Prefix Segment of Node3.

Example 5-107: ISIS LSP originated by Node2

```
 1 RP/0/0/CPU0:xrvr-2#show isis database xrvr-2 verbose
 2
 3 IS-IS 1 (Level-2) Link State Database
 4 LSPID                 LSP Seq Num  LSP Checksum  LSP
Holdtime  ATT/P/OL
 5 xrvr-2.00-00        * 0x0000000f   0x43e8
1187            0/0/0
 6   Area Address:   49.0001
 7   NLPID:          0xcc
 8   Hostname:       xrvr-2
 9   IP Address:     1.1.1.2
10   Router Cap:     1.1.1.2, D:0, S:0
11     Segment Routing: I:1 V:0, SRGB Base: 16000 Range: 8000
12   Metric: 10         IS-Extended xrvr-1.00
13     Interface IP Address: 99.1.2.2
14     Neighbor IP Address: 99.1.2.1
```

```
     ADJ-SID: F:0 B:1 V:1 L:1 S:0 weight:0 Adjacency-sid:30201
     ADJ-SID: F:0 B:0 V:1 L:1 S:0 weight:0 Adjacency-sid:31201
   Metric: 10         IP-Extended 1.1.1.2/32
     Prefix-SID Index: 2, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
   Metric: 0          IP-Extended 1.1.1.3/32
     Prefix-SID Index: 3, Algorithm:0, R:1 N:0 P:1 E:0 V:0 L:0
   Metric: 0          IP-Extended 1.1.1.4/32
     Prefix-SID Index: 4, Algorithm:0, R:1 N:0 P:1 E:0 V:0 L:0
   Metric: 0          IP-Extended 1.1.1.5/32
     Prefix-SID Index: 5, Algorithm:0, R:1 N:0 P:1 E:0 V:0 L:0
   Metric: 0          IP-Extended 2.1.1.4/32
   Metric: 10         IP-Extended 99.1.2.0/24

 Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

Node2 does not redistribute its local loopback0 prefix since that would not redistribute this prefix's associated Prefix-SID. To achieve reachability of this prefix in both domains, Node2 natively advertises its loopback0 prefix in both domains, with its Prefix-SID. Looking back at the OSPF configuration of Node2 in Example 5-101 and Node2's ISIS configuration in Example 5-106, interface loopback0 indeed configured under both routing protocols with the same Prefix-SID 16002.

5.6.1.2 Redistributing between IGP and BGP

The redistribution between OSPF and BGP is in general equivalent to the redistribution between two IGP protocols. OSPF is chosen here as an IGP example, but the redistribution behavior also applies to ISIS. Node3 is the ASBR and it mutually redistributes host prefixes with their Prefix-SIDs between OSPF and BGP. As an example, Node2 originates the prefix 1.1.1.2/32 in OSPF. The RIB entry for prefix 1.1.1.2/32 on Node3 is displayed in Example 5-108. The local label for this prefix is the Prefix-SID label 16002 (line 17).

Example 5-108: RIB entry of Node2's OSPF prefix on Node3

```
 1  RP/0/0/CPU0:xrvr-3#show route 1.1.1.2/32 detail
 2
 3  Routing entry for 1.1.1.2/32
 4    Known via "ospf 2", distance 110, metric 2, labeled SR, type intra area
 5    Installed Aug  1 12:51:31.790 for 00:12:23
 6    Routing Descriptor Blocks
 7      99.2.3.2, from 1.1.1.2, via GigabitEthernet0/0/0/1
 8        Route metric is 2
 9        Label: 0x3 (3)
10        Tunnel ID: None
11        Binding Label: None
12        Extended communities count: 0
13        Path id:1       Path ref count:0
14        NHID:0x3(Ref:3)
15        OSPF area: 0
16    Route version is 0x9 (9)
17    Local Label: 0x3e82 (16002)
18    IP Precedence: Not Set
19    QoS Group ID: Not Set
20    Flow-tag: Not Set
21    Fwd-class: Not Set
22    Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD Type RIB_SVD_TYPE_LOCAL
23    Download Priority 1, Download Version 60
24    No advertising protos.
```

Node3 redistributes OSPF prefix 1.1.1.2/32 with its Prefix-SID into BGP. The prefix with its Prefix-SID is advertised to Node4 using BGP Labeled Unicast (BGP-LU). The BGP configuration of Node3 is displayed in Example 5-109. The output starts with the definition of three route-policies that are used in the BGP configuration (lines 1 to 15). BGP-LU is enabled for IPv4 with the command `allocate-label all` in line 17; with this command BGP allocates a local label for all prefixes it locally originates.

The `address-family ipv4 labeled-unicast` is then enabled under BGP neighbor Node4 (line 22) to advertise labeled prefixes to Node4.

With the network configuration in line 20, BGP advertises prefix 1.1.1.3/32 with a SID index (`label-index`) 3, as set with the route-policy SID(3).

Segment routing is globally enabled with the SRGB [16000-23999] in lines 33 and 34. This configuration enables the SR extensions of BGP-LU where BGP advertises the Prefix-SID attribute with the labeled prefix.

With the redistribution configuration in line 21, BGP redistributes prefixes that are learned by OSPF instance 2. The redistribution is limited to /32 by the route-policy LOOPBACKS.

Example 5-109: BGP configuration on Node3 to redistribute OSPF into BGP

```
 1 route-policy PASS_ALL
 2   pass
 3 end-policy
 4 !
 5 route-policy LOOPBACKS
 6   if destination in (0.0.0.0/0 eq 32) then
 7     pass
 8   else
 9     drop
10   endif
11 end-policy
12 !
13 route-policy SID($SID)
14   set label-index $SID
15 end-policy
16 !
17 router bgp 3
18  bgp router-id 1.1.1.3
19  address-family ipv4 unicast
20   network 1.1.1.3/32 route-policy SID(3)
```

```
  **redistribute ospf 2 route-policy LOOPBACKS**
  allocate-label all
 !
 neighbor 99.3.4.4
  remote-as 4
  description eBGP-LU session to Node4
  address-family ipv4 labeled-unicast
   route-policy PASS_ALL in
   route-policy PASS_ALL out
  !
 !
!
segment-routing
 global-block 16000 23999
!
```

The additional configuration shown in Example 5-110 is required to make forwarding work for BGP-LU learned prefixes. This static route points the BGP nexthop's prefix (99.3.4.4/32) to the outgoing interface. Without this configuration CEF cannot resolve the labels of the BGP-LU routes.

Example 5-110: Static route required to resolve BGP-LU routes

```
router static
 address-family ipv4 unicast
  99.3.4.4/32 GigabitEthernet0/0/0/0
 !
!
```

Node4 receives the BGP-LU update. The output of Example 5-111 shows the BGP table entry for prefix 1.1.1.2/32 as collected on Node4. Node4 receives the Prefix-SID label 16002 for 1.1.1.2/32 (line 14). The BGP update received from Node3 includes the label-index attribute, indicating that prefix 1.1.1.2/32 has a SID with index 2 (`Label Index: 2` in line 18). Node4 is SR enabled and therefore also allocates the Prefix-SID label 16002 as local label (line 6).

Example 5-111: BGP table entry for Node2's prefix on Node4

```
 1| RP/0/0/CPU0:xrvr-4#show bgp ipv4 labeled-unicast 1.1.1.2/32
 2| BGP routing table entry for 1.1.1.2/32
 3| Versions:
 4|   Process           bRIB/RIB  SendTblVer
 5|   Speaker                 35          35
 6|     Local Label: 16002
 7| Last Modified: Aug  1 13:14:20.802 for 00:47:59
 8| Paths: (1 available, best #1)
 9|   Not advertised to any peer
10|   Path #1: Received by speaker 0
11|   Not advertised to any peer
12|   3
13|     99.3.4.3 from 99.3.4.3 (1.1.1.3)
14|       Received Label 16002
15|       Origin incomplete, metric 2, localpref 100, valid,
external, best, group-best
16|       Received Path ID 0, Local Path ID 0, version 35
17|       Origin-AS validity: not-found
18|       Prefix SID Attribute Size: 10
19|       Label Index: 2
```

Node3 also redistributes in the other direction: from BGP into OSPF. BGP on Node4 advertises Node4's loopback prefix 1.1.1.4/32 to Node3 with a Prefix-SID label 16004 (label index 4). The RIB entry of 1.1.1.4/32 on Node3 is displayed in Example 5-112. This RIB entry is installed by `bgp 3`, as shown in line 4. The local label is the Prefix-SID label 16004 (line 16).

Example 5-112: RIB entry of Node4's BGP prefix on Node3

```
1| RP/0/0/CPU0:xrvr-3#show route 1.1.1.4/32 detail
2|
3| Routing entry for 1.1.1.4/32
4|   Known via "bgp 3", distance 20, metric 0, [ei]-bgp,
labeled unicast (3107), labeled SR
5|   Tag 4, type external
6|   Installed Aug  1 13:14:20.716 for 00:55:14
7|   Routing Descriptor Blocks
```

```
      99.3.4.4, from 99.3.4.4, BGP external
        Route metric is 0
        Label: 0x100004 (1048580)
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        NHID:0x0(Ref:0)
    Route version is 0x2 (2)
    Local Label: 0x3e84 (16004)
    IP Precedence: Not Set
    QoS Group ID: Not Set
    Flow-tag: Not Set
    Fwd-class: Not Set
    Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type
RIB_SVD_TYPE_LOCAL
    Download Priority 4, Download Version 75
    No advertising protos.
```

OSPF on Node3 is configured to redistribute BGP into OSPF, see Example 5-113.

Example 5-113: OSPF configuration on Node3 to redistribute BGP into OSPF

```
route-policy LOOPBACKS
  if destination in (0.0.0.0/0 eq 32) then
    pass
  else
    drop
  endif
end-policy
!
router ospf 2
 segment-routing mpls
 redistribute bgp 3 route-policy LOOPBACKS
 area 0
  interface Loopback0
   passive enable
   prefix-sid absolute 16003
  !
  interface Loopback2
   passive enable
  !
```

```
   interface GigabitEthernet0/0/0/1
    network point-to-point
   !
  !
 !
```

OSPF on Node3 redistributes prefix 1.1.1.4/32 and advertises it in an External (type 5) LSA. This External LSA is displayed in Example 5-114. Notice the `External Route Tag: 4` on line 21. OSPF tags the prefix with the BGP peer's AS number 4. This is something unrelated to SR, but automatically done by default when redistributing BGP into OSPF. This tag is used for loop prevention.

Example 5-114: OSPF External LSA originated by Node3

```
 1 RP/0/0/CPU0:xrvr-3#show ospf database external self-
originate
 2
 3
 4              OSPF Router with ID (1.1.1.3) (Process ID 2)
 5
 6                  Type-5 AS External Link States
 7
 8   LS age: 1700
 9   Options: (No TOS-capability, DC)
10   LS Type: AS External Link
11   Link State ID: 1.1.1.4 (External Network Number)
12   Advertising Router: 1.1.1.3
13   LS Seq Number: 80000002
14   Checksum: 0xf8a8
15   Length: 36
16   Network Mask: /32
17         Metric Type: 2 (Larger than any link state path)
18         TOS: 0
19         Metric: 1
20         Forward Address: 0.0.0.0
21         External Route Tag: 4
22
```

OSPF advertises the Prefix-SID of the redistributed prefix 1.1.1.4/32 in an Extended Prefix LSA, as displayed in Example 5-115. It is a type 11 LSA (line 6), which is an Opaque LSA that is flooded throughout the AS. The Extended Prefix TLV in this LSA identifies the prefix to which it applies (lines 19 to 23). It is an external (`Route-type: 5`) prefix 1.1.1.4/32. None of the flags is set (`Flags: 0x0`). The Prefix-SID sub-TLV (lines 25 to 29) identifies the Prefix-SID and its attributes. The SID index is 4 and only the PHP-off flag is set (NP:1).

Example 5-115: Extended Prefix LSA originated by Node3

```
 1| RP/0/0/CPU0:xrvr-3#show ospf database opaque-as 1.1.1.4/32
 2|
 3|
 4|             OSPF Router with ID (1.1.1.3) (Process ID 2)
 5|
 6|                 Type-11 Opaque Link AS Link States
 7|
 8|   LS age: 1852
 9|   Options: (No TOS-capability, DC)
10|   LS Type: Opaque AS Link
11|   Link State ID: 7.0.0.1
12|   Opaque Type: 7
13|   Opaque ID: 1
14|   Advertising Router: 1.1.1.3
15|   LS Seq Number: 80000003
16|   Checksum: 0x9ee8
17|   Length: 44
18|
19|     Extended Prefix TLV: Length: 20
20|       Route-type: 5               !! External prefix
21|       AF        : 0
22|       Flags     : 0x0             !! A:0; N:0
23|       Prefix    : 1.1.1.4/32
24|
25|       SID sub-TLV: Length: 8
26|         Flags     : 0x40          !! (NA); NP:1; M:0; E:0;
V:0; L:0
27|         MTID      : 0
28|         Algo      : 0             !! SPF
```

```
        SID Index : 4
```

To distribute its local loopback prefix in both OSPF and BGP domains, Node3 configures both protocols to advertise the prefix with its Prefix-SID. For OSPF this is done by configuring loopback0 under OSPF and configuring a Prefix-SID for it, see configuration in Example 5-113. In BGP this is done by advertising 1.1.1.3/32 in address-family ipv4 labeled-unicast and set the label-index attribute value to the SID index, see Example 5-109.

5.6.2 OSPF NSSA

Since OSPF advertises external prefixes differently when an Autonomous System Boundary Router (ASBR) is located in a Not-So-Stubby area (NSSA), a NSSA is added to the example network topology to illustrate the behavior. See Figure 5-38. The OSPF domain now consists of two areas: the backbone area (Area 0) and a Not-so-Stubby area (NSSA) (Area 1). The Node5 in the NSSA is connected to the BGP domain and redistributes BGP prefixes into the OSPF domain. Node5 is an ASBR. If an ASBR is located in an NSSA then it originates NSSA External (type 7) LSAs for the external prefixes instead of External (type 5) LSAs. The ABR Node3 then translates these NSSA External LSAs into an External LSAs and originates the External LSAs in the backbone Area 0.

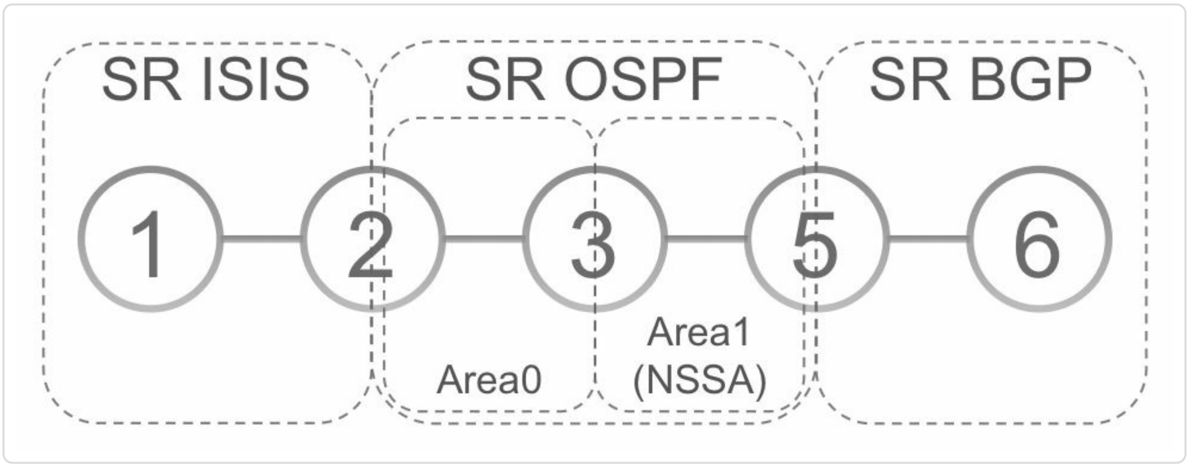


Figure 5-38: Redistribution with NSSA in OSPF domain

Node6 advertises prefix 1.1.1.6/32 with Prefix-SID 16006 in BGP-LU to Node5. Node5 installs the prefix in RIB. The RIB entry of 1.1.1.6/32 on Node5 is shown in Example 5-116. The entry is installed by `bgp 5` (line 4). The local label for this prefix is the Prefix-SID label 16006 (line 16).

Example 5-116: RIB entry of Node6's BGP prefix on Node5

```
 1 RP/0/0/CPU0:xrvr-5#show route 1.1.1.6/32 detail
 2
 3 Routing entry for 1.1.1.6/32
 4   Known via "bgp 5", distance 20, metric 0, [ei]-bgp,
labeled unicast (3107), labeled SR
 5   Tag 6, type external
 6   Installed Aug  1 15:25:09.298 for 00:18:10
 7   Routing Descriptor Blocks
 8     99.5.6.6, from 99.5.6.6, BGP external
 9       Route metric is 0
10       Label: 0x100004 (1048580)
11       Tunnel ID: None
12       Binding Label: None
13       Extended communities count: 0
14       NHID:0x0(Ref:0)
15   Route version is 0x2 (2)
16   Local Label: 0x3e86 (16006)
17   IP Precedence: Not Set
18   QoS Group ID: Not Set
19   Flow-tag: Not Set
```

```
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type
RIB_SVD_TYPE_LOCAL
  Download Priority 4, Download Version 85
  No advertising protos.
```

OSPF on Node5 is configured to redistribute the BGP host prefixes, as shown in the configuration in Example 5-117. Area 1 is an NSSA, as shown in line 13.

Example 5-117: OSPF configuration on Node5 to redistribute BGP into OSPF

```
 1 route-policy LOOPBACKS
 2   if destination in (0.0.0.0/0 eq 32) then
 3     pass
 4   else
 5     drop
 6   endif
 7 end-policy
 8 !
 9 router ospf 2
10  segment-routing mpls
11  redistribute bgp 5 route-policy LOOPBACKS
12  area 1
13   nssa
14   interface Loopback0
15    passive enable
16    prefix-sid absolute 16005
17   !
18   interface GigabitEthernet0/0/0/0
19    network point-to-point
20   !
21  !
22 !
```

Node5 advertises the redistributed prefix 1.1.1.6/32 in an OSPF NSSA External (type 7) LSA. This LSA is displayed in Example 5-118.

Example 5-118: NSSA External LSA originated by Node5

```
RP/0/0/CPU0:xrvr-5#show ospf database nssa-external self-
originate


            OSPF Router with ID (1.1.1.5) (Process ID 2)

                Type-7 AS External Link States (Area 1)

  LS age: 1410
  Options: (No TOS-capability, Type 7/5 translation, DC)
  LS Type: AS External Link
  Link State ID: 1.1.1.6 (External Network Number)
  Advertising Router: 1.1.1.5
  LS Seq Number: 80000001
  Checksum: 0xd4b5
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 1
        Forward Address: 1.1.1.5
        External Route Tag: 6
```

Node5 also advertises the Prefix-SID of this redistributed prefix in an Extended Prefix LSA, as shown in Example 5-119. This Extended Prefix LSA is a type 10 LSA; it is an Opaque LSA with area flooding scope. It has area flooding-scope since the NSSA External LSA also has area flooding-scope. This is in contrast with the Extended Prefix LSA that is advertised for a prefix advertised in an External LSA, which is distributed AS-wide.

The Extended Prefix TLV in the LSA identifies the prefix (lines 19 to 23). It is an NSSA external prefix (`Route-type: 7`) with no flags set (`Flags: 0x0`). The Prefix-SID and its attributes are indicated in the

Prefix-SID sub-TLV. The SID index is 6 and only the PHP-off flag is set (NP:1).

Example 5-119: Extended Prefix LSA originated by Node5

```
RP/0/0/CPU0:xrvr-5#show ospf database opaque-area 1.1.1.6/32 self-originate


            OSPF Router with ID (1.1.1.5) (Process ID 2)

                Type-10 Opaque Link Area Link States (Area 1)

  LS age: 1888
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.2
  Opaque Type: 7
  Opaque ID: 2
  Advertising Router: 1.1.1.5
  LS Seq Number: 80000001
  Checksum: 0xf888
  Length: 44

    Extended Prefix TLV: Length: 20
      Route-type: 7              !! NSSA External prefix
      AF        : 0
      Flags     : 0x0            !! A:0; N:0
      Prefix    : 1.1.1.6/32

      SID sub-TLV: Length: 8
        Flags     : 0x40         !! (NA); NP:1; M:0; E:0; V:0; L:0
        MTID      : 0
        Algo      : 0            !! SPF
        SID Index : 6

```

5.7 Summary

- The classic MPLS control-plane uses LDP to distribute the labels associated with the IGP prefixes. This is a redundant and complex design decision. SR eliminates this complexity.
- In SR, straightforward IGP extensions are implemented to distribute these labels natively in the IGP.
- Eliminating LDP means less code to maintain and qualify, less software bugs to experience and troubleshoot, and less performance drain on the distributing routing system.
- Eliminating LDP means simpler SDN operation as the controller only needs to retrieve the IGP link-state topology to infer the topology and the related segments.
- Eliminating LDP means eliminating the infamous LDP/synch problem (IETF RFC 5443).
- This section details the SR IGP extensions and how they are used within the network to distribute the SR node capabilities and install the prefix and adjacency segments in the MPLS forwarding plane. The control of penultimate-hop processing/explicit-null is analyzed.
- Prefix-SIDs are propagated with their associated prefix between levels and areas.
- Prefix-SIDs are redistributed with its prefix when redistributing prefixes between SR IGPs or between SR IGP and SR BGP.

5.8 References

- [draft-ietf-isis-segment-routing-extensions] Previdi, S., Ed., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions (work in progress), June 2016, https://tools.ietf.org/html/draft-ietf-isis-segment-routing-extensions.
- [draft-ietf-ospf-ospfv3-lsa-extend] Lindem, A., Mirtorabi, S., Roy, A., and F. Baker, "OSPFv3 LSA Extendibility", draft-ietf-ospf-ospfv3-lsa-extend-10 (work in progress), May 2016, https://tools.ietf.org/html/draft-ietf-ospf-ospfv3-lsa-extend.
- [draft-ietf-ospf-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-09 (work in progress), July 2016, https://tools.ietf.org/html/draft-ietf-ospf-segment-routing-extensions.
- [draft-ietf-spring-conflict-resolution] Ginsberg, L., Psenak, P., Previdi, S., and M. Pilka, "Segment Routing Conflict Resolution", draft-ietf-spring-conflict-resolution (work in progress), June 2016, https://tools.ietf.org/html/draft-ietf-spring-conflict-resolution.
- [draft-ppsenak-ospf-te-link-attr-reuse] Peter Psenak, Acee Lindem, Les Ginsberg, Wim Henderickx, Jeff Tantsura, Hannes Gredler, "OSPFv2 Link Traffic Engineering (TE) Attribute Reuse", draft-ppsenak-ospf-te-link-attr-reuse (work in progress), August 2016, https://tools.ietf.org/html/draft-ppsenak-ospf-te-link-attr-reuse.
- [ietf-ospf-ospfv3-segment-routing-extensions] 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 (work in progress), July 2016, https://tools.ietf.org/html/draft-ietf-ospf-ospfv3-segment-routing-extensions.

- [RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, DOI 10.17487/RFC1195, December 1990, http://www.rfc-editor.org/info/rfc1195.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, http://www.rfc-editor.org/info/rfc2328.
- [RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, http://www.rfc-editor.org/info/rfc3630.
- [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, http://www.rfc-editor.org/info/rfc4915.
- [RFC4971] Vasseur, JP., Ed., Shen, N., Ed., and R. Aggarwal, Ed., "Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information", RFC 4971, DOI 10.17487/RFC4971, July 2007, http://www.rfc-editor.org/info/rfc4971.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, http://www.rfc-editor.org/info/rfc5120.
- [RFC5250] Berger, L., Bryskin, I., Zinin, A., and R. Coltun, "The

OSPF Opaque LSA Option", RFC 5250, DOI 10.17487/RFC5250, July 2008, http://www.rfc-editor.org/info/rfc5250.

- [RFC5302] Li, T., Smit, H., and T. Przygienda, "Domain-Wide Prefix Distribution with Two-Level IS-IS", RFC 5302, DOI 10.17487/RFC5302, October 2008, http://www.rfc-editor.org/info/rfc5302.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, http://www.rfc-editor.org/info/rfc5305.
- [RFC5443] Jork, M., Atlas, A., and L. Fang, "LDP IGP Synchronization", RFC 5443, DOI 10.17487/RFC5443, March 2009, http://www.rfc-editor.org/info/rfc5443.
- [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, http://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, http://www.rfc-editor.org/info/rfc7770.
- [RFC7794] Ginsberg, L., Ed., Decraene, B., Previdi, S., Xu, X., and U. Chunduri, "IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability", RFC 7794, DOI 10.17487/RFC7794, March 2016, http://www.rfc-editor.org/info/rfc7794.

[1] [draft-ietf-spring-conflict-resolution]

[2] Another computation may be needed if the SRGB consists of multiple label ranges. See chapter 4, “Management of the SRGB”.

6 SEGMENT ROUTING BGP CONTROL PLANE

It may seem that this book focuses on the Segment Routing IGP control plane. However, SR is not constrained to using the IGP control plane. This chapter discusses using the Border Gateway Protocol (BGP) to distribute SR information.

SR BGP is not limited to a specific SR data plane implementation, and can be applied to the MPLS and IPv6 data planes, similar to SR IGP. However, this chapter focuses on the SR MPLS application of SR BGP like most of this book.

REMINDER: BGP Prefix Segment

(from chapter 2, “Fundamentals of Segment Routing”)

A BGP Prefix Segment is a segment that is attached to a BGP prefix. A BGP Prefix-Segment Identifier (Prefix-SID) is the identifier of the BGP Prefix Segment. A BGP Prefix-SID is a Global Segment and identifies an instruction to forward via the ECMP-aware BGP best-path to the associated prefix. The Prefix-SID consists of a label for the SR MPLS application, while the Prefix-SID consists of an IPv6 prefix for the SR IPv6 data plane implementation (SRv6).

When introducing the SR Fundamentals in chapter 2, “Fundamentals of Segment Routing”, we also briefly described the BGP Peer Segments and extensions to BGP Link State (BGP-LS) protocol for signaling SR information. Both of these elements come in to play with SR Traffic Engineering and hence will be covered in the next part of this book. In this chapter we will focus on the BGP Prefix Segment; how BGP is extended to signal such segments and their related forwarding entries.

The SR BGP concepts and mechanisms extend to the designs in Service Provider or Enterprise networks that today use BGP (and as well BGP Labeled Unicast) for their transport signaling. SR BGP can for example be used to connect different SR IGP domains, as discussed in the section about redistribution of the SR IGP chapter chapter 5, “Segment Routing IGP Control Plane”; in this chapter we will start by delving deeper into the BGP aspects of this solution. However, later in this chapter we will focus more on the Data Center application because those deployments were the key initial drivers for SR BGP. In these Data Center designs, BGP is the IGP and hence it serves as a very good example. This should not create a misconception of SR BGP being mostly applicable for Data Center networks. We will also look at an extension of the Seamless MPLS architecture with SR BGP that we touch upon when discussing large scale interconnect solutions in chapter chapter 10, “Large Scale Interconnect with Segment Routing”.

6.1 BGP Labeled Unicast

BGP Labeled Unicast (BGP-LU), as specified in IETF RFC 3107, is used to distribute MPLS labels for prefixes in BGP. BGP-LU is therefore also known by the name “BGP RFC 3107”. RFC 3107 specifies how one or more labels for a prefix can be carried in the BGP update message using the Multiprotocol Extensions of BGP (MBGP); the labels are advertised as part of the Network Layer Reachability Information (NLRI). Cisco IOS XR currently supports advertising a single label for a prefix in BGP-LU.

REMINDER: Multi-protocol BGP (MBGP)

Standard BGP only supports IPv4 Unicast addresses. MBGP is an extension to BGP that enables distributing different types of addresses (known as address-families). The standard BGP (IETF RFC 4271) Update message format only supports IPv4 prefixes; therefore new path attributes have been defined to support non-IPv4 network protocols. Multi-protocol BGP (MBGP) specified in IETF RFC 4760 enables BGP to support different address-families. An "Address Family Identifier" (AFI) and a "Subsequent Address Family Identifier" (SAFI) identifies each address family. One of the supported address-families is IPv4 labeled-unicast, identified by an AFI value 1 for IPv4 and a SAFI value 4. Before the introduction of MBGP in RFC 4760, no address-families had to be configured for BGP since only IPv4 was supported.

IETF RFC 4760 introduces new optional non-transitive attributes: MP_REACH_NLRI and MP_UNREACH_NLRI. IETF RFC 4760 specifies that MP_REACH_NLRI “is used to carry the set of reachable destinations together with the next hop information to be used for forwarding to these destinations.” and MP_UNREACH_NLRI “is used to carry the set of unreachable destinations.”

When a node receives a BGP-LU Update for a prefix with a label, it installs that label in the forwarding table as the outgoing label for that prefix. The node also allocates a local label for that prefix, which it installs as an incoming label in the MPLS forwarding entry. When that node then advertises this prefix to another node and it advertises this prefix with itself as nexthop, then it includes its own local label in the BGP-LU Update message.

When a node advertises a *local* prefix in BGP-LU to a BGP peer, it advertises the Implicit-null label value 3 with it by default. This is for the Penultimate Hop Popping functionality. The neighbor that receives this update installs a pop MPLS forwarding entry for that prefix; it will pop the label before forwarding the packet. The node also allocates a local label for the prefix, installs it in the MPLS forwarding table as an incoming label, and advertises that label to other nodes.

BGP-LU signaling is applicable for both external-BGP (EBGP) and internal-BGP (IBGP) sessions. The EBGP sessions could be between ASs belonging to different operators but often also between multiple IGP domains or networks/divisions belonging to the same operator.

We will use the topology in Figure 6-1 to illustrate BGP-LU in a multi-AS network. The network consists of two Autonomous Systems (ASs). AS1 is on the left, AS2 on the right. Node4 and Node5 are on the border between the ASs; these are AS Border Routers (ASBRs). One ASBR belongs to AS1, the other one belongs to AS2. The ASBRs are connected via a peering link.

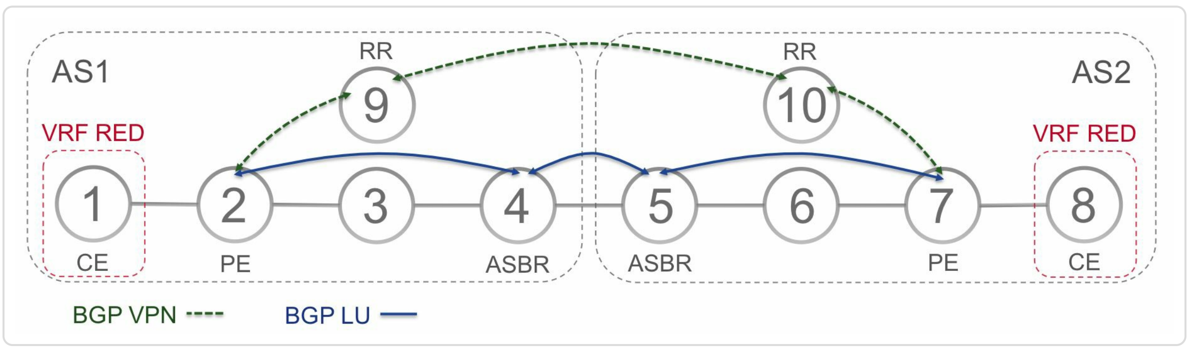


Figure 6-1: Inter-AS example topology

A L3VPN service is activated between CE Node1 and CE Node8. These CEs are connected to PEs Node2 and Node7 respectively. A VRF named “RED” is instantiated on both PEs.

The challenge is to connect Node1 to Node8 across the AS boundary. IETF RFC 4364, “BGP/MPLS IP Virtual Private Networks (VPNs)”, describes three ways to handle this case, listed as a), b), and c). Therefore, these three options are known as inter-AS option A, B, and C.

The example only discusses inter-AS option C. In this model, the L3VPN prefixes and labels are exchanged between the PEs in the two ASs using EBGP. For scalability, this exchange typically happens over a multi-hop EBGP session between a Route Reflector (RR) in one AS and a RR in the other AS. The RRs in this example are Node9 for As1 and Node10 for AS2. The RRs preserve the nexthop of the routes they advertise over this BGP session (the `next-hop-unchanged` functionality). The L3VPN BGP sessions are indicated in Figure 6-1 with the label “BGP VPN”.

To establish the inter-AS BGP session between the RRs, reachability between them is required. Inter-AS option C also requires that the loopback prefixes of the PEs are reachable from the other AS. This is required to provide a continuous Label Switched Path (LSP) between the PEs to carry the L3VPN service traffic. Since the RRs reflect the routes with an unchanged nexthop, for example Node2 receives a L3VPN route with Node7 as BGP nexthop. Hence Node7 must be reachable (with label) from Node2.

The regular label distribution mechanisms (e.g. LDP or SR IGP as described in chapter 5, “Segment Routing IGP Control Plane”) can be used within each AS, while BGP Labeled Unicast (BGP-LU) can be used to exchange MPLS labels over the inter-AS link.

Two options are available to provide an inter-AS LSP between the PEs:

- Mutually redistribute the PE loopback prefixes with their labels between BGP and IGP (BGP<→IGP)
- Distribute the PE loopback prefixes with their labels in BGP-LU

Only the second option (BGP-LU) is discussed here. Mutual redistribution between IGP and BGP with Prefix-SIDs is discussed in chapter 5, "Segment Routing IGP Control Plane".

The configuration of Node7 is shown in Example 6-1.

Example 6-1: BGP configuration example – PE Node7

```
route-policy bgp_in
  pass
end-policy
!
route-policy bgp_out
  pass
end-policy
!
router bgp 2
 bgp router-id 1.1.1.7
 address-family ipv4 unicast
  network 1.1.1.7/32
  allocate-label all
 !
 address-family vpnv4 unicast
 !
 neighbor 1.1.1.5
  remote-as 2
  description iBGP peer xrvr-5
  update-source Loopback0
  address-family ipv4 labeled-unicast
```

```
   !
  !
  neighbor 1.1.1.10
   remote-as 2
   description iBGP peer xrvr-10
   update-source Loopback0
   address-family vpnv4 unicast
  !
 !
  vrf RED
   rd auto
   address-family ipv4 unicast
   !
   neighbor 99.7.8.8
    remote-as 108
    description eBGP peer xrvr-8
    address-family ipv4 unicast
     route-policy bgp_in in
     route-policy bgp_out out
    !
   !
  !
 !
```

Node7 advertises its loopback prefix 1.1.1.7/32 with a label 3 (implicit-null) to Node5. This direct BGP-LU session between PE and ASBR is for illustration purposes only. In reality a RR will be used, the same RR as for the L3VPN service or a different one. Example 6-2 shows the BGP table entry of 1.1.1.7/32 on Node5. The BGP nexthop is 1.1.1.7 and the outgoing label is 3 (implicit-null).

Example 6-2: BGP-LU prefix 1.1.1.7/32 on Node5

```
RP/0/0/CPU0:xrvr-5#show bgp ipv4 labeled-unicast 1.1.1.7/32
BGP routing table entry for 1.1.1.7/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                  4           4
    Local Label: 95003
Last Modified: Sep 29 10:11:55.855 for 04:31:00
```

```
 Paths: (1 available, best #1)
   Advertised to peers (in unique update groups):
     99.4.5.4
   Path #1: Received by speaker 0
   Advertised to peers (in unique update groups):
     99.4.5.4
   Local
     1.1.1.7 (metric 20) from 1.1.1.7 (1.1.1.7)
       Received Label 3
       Origin IGP, metric 0, localpref 100, valid, internal,
 best, group-best
       Received Path ID 0, Local Path ID 0, version 4
```

The BGP configuration of ASBR Node4 is shown in Example 6-3. Node4 has an EBGP session to Node5 and IBGP sessions to Node7 and Node10. `Next-hop-self` is configured on these IBGP sessions; see line 19. To resolve the labeled BGP cef entries, a static route to the BGP nexthop is needed; see line 41.

Example 6-3: BGP configuration example – ASBR Node4

```
 1| route-policy bgp_in
 2|   pass
 3| end-policy
 4| !
 5| route-policy bgp_out
 6|   pass
 7| end-policy
 8| !
 9| router bgp 2
10|  bgp router-id 1.1.1.5
11|  !
12|  address-family ipv4 unicast
13|   allocate-label all
14|  !
15|  neighbor-group IBGP
16|   remote-as 2
17|   update-source Loopback0
18|   address-family ipv4 labeled-unicast
19|    next-hop-self
20|   !
```

```
 !
 neighbor 1.1.1.7
  use neighbor-group IBGP
  description iBGP peer xrvr-7
 !
 neighbor 1.1.1.10
  use neighbor-group IBGP
  description iBGP peer xrvr-10
 !
 neighbor 99.4.5.4
  remote-as 1
  description eBGP peer xrvr-4
  address-family ipv4 labeled-unicast
   route-policy bgp_in in
   route-policy bgp_out out
  !
 !
!
router static
 address-family ipv4 unicast
  99.4.5.4/32 GigabitEthernet0/0/0/1
 !
!
```

For example, Node5 advertises prefix 1.1.1.7/32 with label 90507 to Node4. See the BGP table entry of 1.1.1.7/32 on Node4 in Example 6-4. The BGP nexthop is 99.4.5.5, which is the interface address of Node5, and the `Received Label` is 90507. Node4 has allocated a `Local Label` 90407 for this prefix.

Example 6-4: BGP-LU prefix 1.1.1.7/32 on Node4

```
RP/0/0/CPU0:xrvr-4#show bgp ipv4 labeled-unicast 1.1.1.7/32
BGP routing table entry for 1.1.1.7/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                  4           4
    Local Label: 90407
Last Modified: Sep 29 10:13:30.899 for 04:44:01
Paths: (1 available, best #1)
  Advertised to update-groups (with more than one peer):
    0.3
  Path #1: Received by speaker 0
  Advertised to update-groups (with more than one peer):
    0.3
  2
    99.4.5.5 from 99.4.5.5 (1.1.1.5)
      Received Label 90507
      Origin IGP, localpref 100, valid, external, best, group-
best
      Received Path ID 0, Local Path ID 0, version 4
      Origin-AS validity: not-found
```

Node4 then advertises this prefix with its label to Node2. The BGP table entry on Node2 is shown in Example 6-5. The BGP nexthop is 1.1.1.4 and the `Received Label` is 90407.

Example 6-5: BGP-LU prefix 1.1.1.7/32 on Node4

```
RP/0/0/CPU0:xrvr-2#show bgp ipv4 labeled-unicast 1.1.1.7/32
BGP routing table entry for 1.1.1.7/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 18          18
    Local Label: 90207
Last Modified: Sep 29 10:13:30.813 for 04:49:13
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  2
    1.1.1.4 (metric 20) from 1.1.1.4 (1.1.1.4)
      Received Label 90407
      Origin IGP, localpref 100, valid, internal, best, group-
best
      Received Path ID 0, Local Path ID 0, version 18
```

LDP is currently enabled in each AS to provide intra-AS MPLS transport. The LDP labels are used to transport the packets within the AS. Figure 6-2 shows the BGP-LU label exchange (a), the LDP label exchange (b) and the transport label stack (c). On Node5, the BGP label (90507) is popped and the LDP label that Node6 advertised for 1.1.1.7/32 (90607) is pushed on the packet, which results in a swap of label 90507 with 90607.

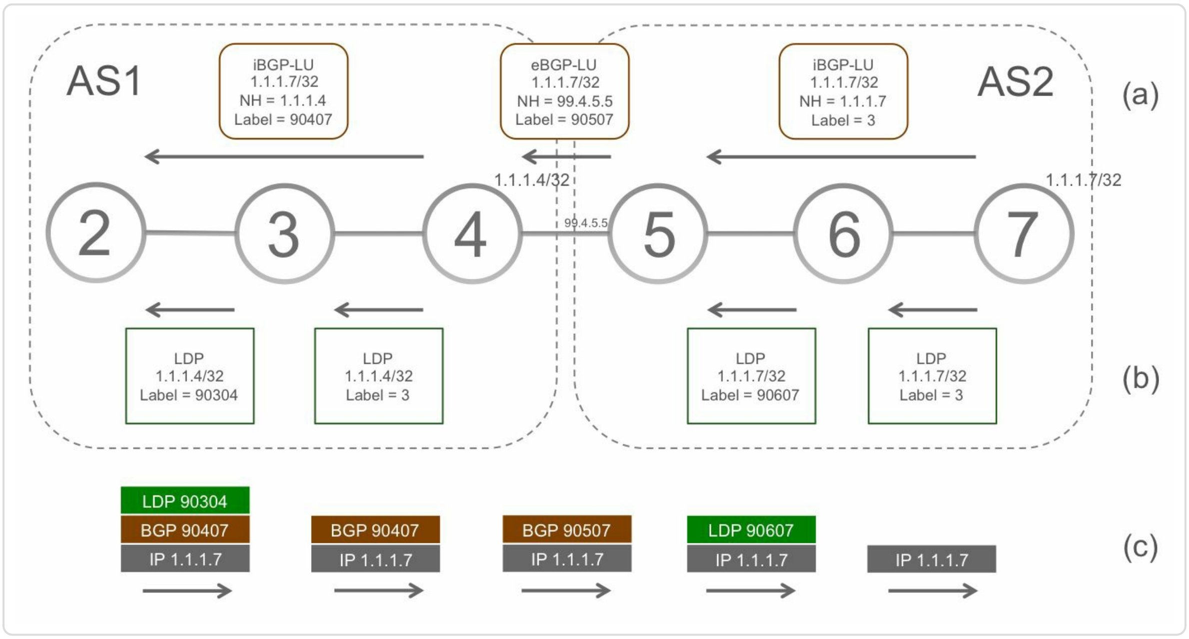


Figure 6-2: Inter-AS Label switched path between PEs – no SR

Executing a traceroute on Node2 to Node7 results in the output shown in Example 6-6.

Example 6-6: Traceroute example on PE Node2 to PE Node7

```
RP/0/0/CPU0:xrvr-2#traceroute 1.1.1.7 source 1.1.1.2

Type escape sequence to abort.
Tracing the route to 1.1.1.7

 1  99.2.3.3 [MPLS: Labels 90304/90407 Exp 0] 19 msec  19 msec
9 msec
 2  99.3.4.4 [MPLS: Label 90407 Exp 0] 9 msec  9 msec  9 msec
 3  99.4.5.5 [MPLS: Label 90507 Exp 0] 9 msec  9 msec  9 msec
 4  99.5.6.6 [MPLS: Label 90607 Exp 0] 9 msec  9 msec  9 msec
 5  99.6.7.7 9 msec  9 msec  9 msec
```

Figure 6-3: Inter-AS end-to-end Label switched path between CEs – no SR

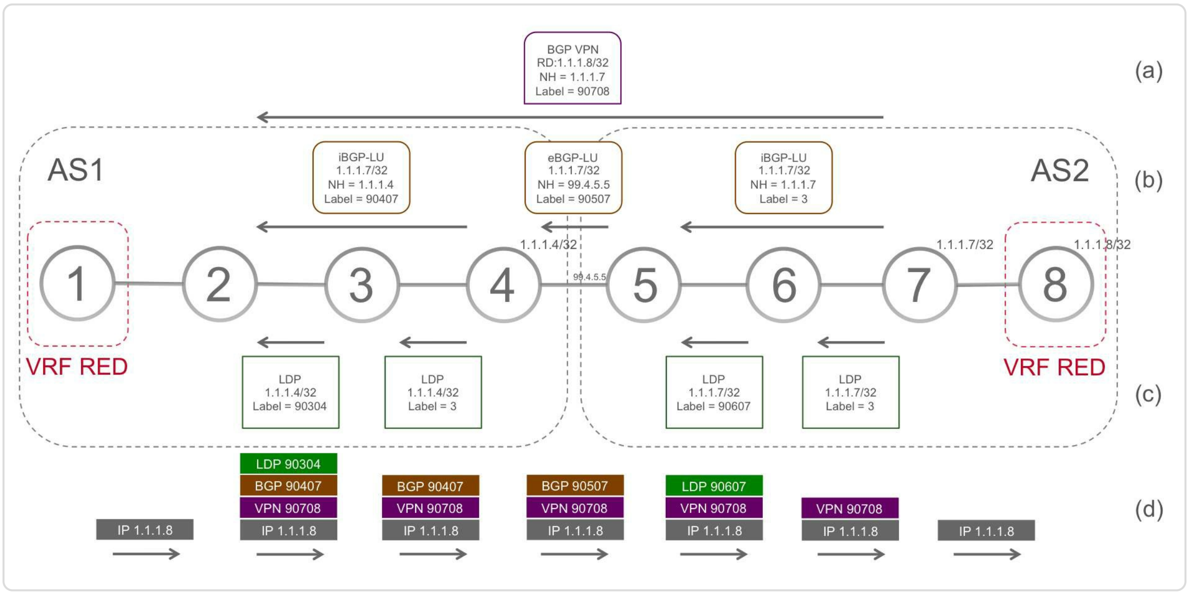


Figure 6-3: Traceroute example on CE Node1 to CE Node8 – no SR

Example 6-7: Traceroute example on CE Node1 to CE Node8 – no SR

```
RP/0/0/CPU0:xrvr-1#traceroute 1.1.1.8

Type escape sequence to abort.
Tracing the route to 1.1.1.8

 1   99.1.2.2 9 msec  0 msec  0 msec
 2   99.2.3.3 [MPLS: Labels 90304/90407/90708 Exp 0] 19 msec  19
msec  9 msec
 3   99.3.4.4 [MPLS: Labels 90407/90708 Exp 0] 9 msec  9 msec  9
msec
 4   99.4.5.5 [MPLS: Labels 90507/90708 Exp 0] 9 msec  9 msec  9
msec
 5   99.5.6.6 [MPLS: Labels 90607/90708 Exp 0] 9 msec  9 msec  9
msec
 6   99.6.7.7 [MPLS: Label 90708 Exp 0] 9 msec  9 msec  9 msec
 7   99.7.8.8 9 msec  9 msec  9 msec
```

6.2 BGP Prefix-SID

SR BGP uses the SR architectural concepts, equivalent to SR IGP, but applied to the BGP control plane. Prefix Segments are Global Segments. For the SR MPLS implementation Prefix-SIDs are distributed as globally unique SID indexes pointing to Prefix-SID labels in the SRGB. The Prefix-SID label value is then computed by adding the SID index to the SRGB Base label value.

The previous sections showed how BGP-LU is used to distribute prefix reachability information and MPLS labels for inter-AS connectivity. SR extends BGP-LU to advertise BGP Prefix-SIDs; the advertising node attaches the Prefix-SID of a prefix to the BGP-LU prefix advertisement. The Prefix-SID is configured on the node that originates the prefix or is derived from another protocol if the prefix is redistributed. The Prefix-SID is included as an attribute in the BGP update message (more details will follow in section 6.3, “BGP Prefix-SID Advertisement”).

The same topology as in the previous section is now discussed with SR BGP enabled. See Figure 6-4. All nodes in the topology have SR BGP enabled, using the same SRGB [16000-23999]. A Prefix-SID with SID index 7 is configured on Node7 for its loopback prefix 1.1.1.7/32. The SR BGP configuration of Node7 is shown in Example 6-8. First an SRGB must be configured globally. In this example, the default SRGB [16000-23999] is configured (lines 49 and 50). A route-policy `SID()` is attached to the `network 1.1.1.7/32` statement (line 16). This route-policy is specified in lines 9 to 11 and sets the `label-index` to the parameter that is configured when applying the route-policy, 7 in this example. The rest of the BGP configuration is the same as for classic BGP-LU. SR IGP is also configured on the nodes, but that configuration is not shown here.

Example 6-8: SR BGP configuration example – PE Node7

```
route-policy bgp_in
  pass
end-policy
!
route-policy bgp_out
  pass
end-policy
!
route-policy SID($SID)
  set label-index $SID
end-policy
!
router bgp 2
 bgp router-id 1.1.1.7
 address-family ipv4 unicast
  network 1.1.1.7/32 route-policy SID(7)
  allocate-label all
 !
 address-family vpnv4 unicast
 !
 neighbor 1.1.1.5
  remote-as 2
  description iBGP peer xrvr-5
  update-source Loopback0
  address-family ipv4 labeled-unicast
  !
 !
 neighbor 1.1.1.10
  remote-as 2
  description iBGP peer xrvr-10
  update-source Loopback0
  address-family vpnv4 unicast
 !
!
 vrf RED
  rd auto
  address-family ipv4 unicast
  !
```

```
   neighbor 99.7.8.8
    remote-as 108
    description eBGP peer xrvr-8
    address-family ipv4 unicast
     route-policy bgp_in in
     route-policy bgp_out out
    !
   !
  !
 !
 segment-routing
  global-block 16000 23999
 !
```

On the ASBRs, only the global SRGB [16000-23999] is configured, which enables BGP to allocate Prefix-SID labels when an advertisement with a Prefix-SID attribute is received.

Since prefix 1.1.1.7/32 is local to Node7, it advertises this prefix to Node5 in BGP-LU with label value 3 (implicit-null). The BGP nexthop for this prefix is the loopback address 1.1.1.7. Node7 also adds the BGP Prefix-SID attribute to the update message, specifying Prefix-SID index 7. Notice that both a label value and a SID index are advertised with the prefix (Figure 6-4 (a)).

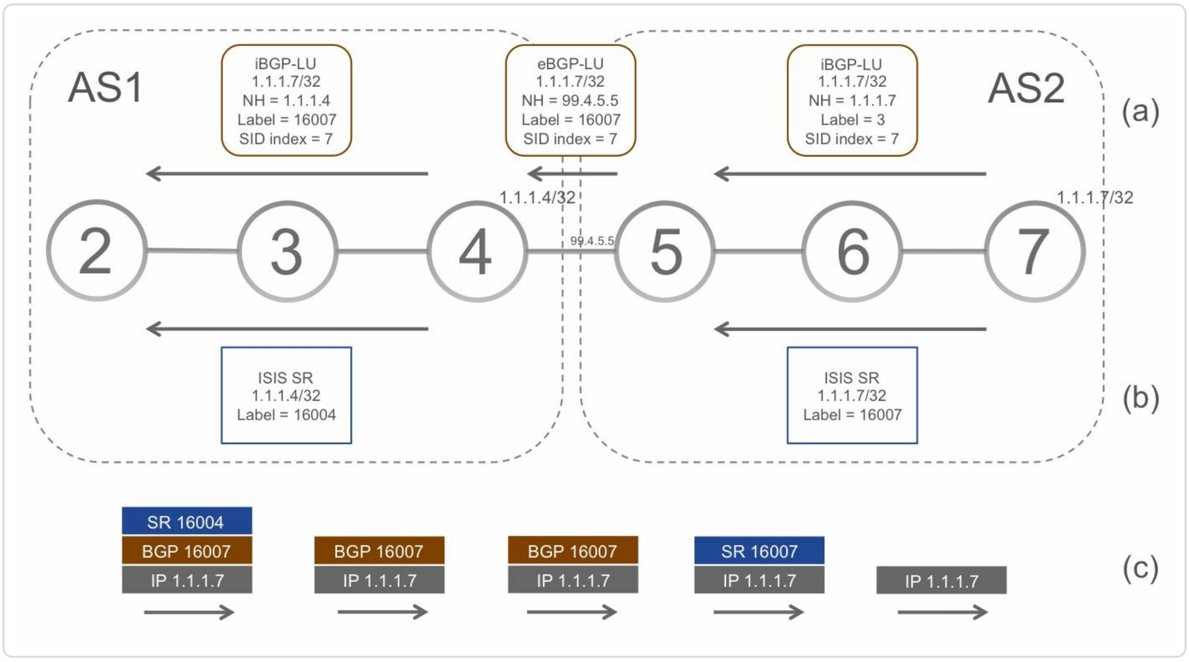


Figure 6-4: Inter-AS Label switched path between PEs – SR

When Node5 receives the prefix update, it allocates a local label for prefix 1.1.1.7/32. This local label is not allocated from the dynamic label range as with regular BGP-LU, but it is allocated from the SRGB, based on the Prefix-SID index attribute. Node5 allocates local label 16007 (= 16000 + 7) for this prefix and then re-advertises the prefix with label 16007 and SID index 7 to its EBGP-LU neighbor Node4. The BGP Prefix-SID attribute is re-advertised unchanged with the prefix. Node4 changes the BGP nexthop for this prefix to its own BGP session address 99.4.5.5 (default EBGP "next-hop-self" behavior).

Node4 does the same operations and eventually Node2 learns the prefix 1.1.1.7/32 with label 16007 and Prefix-SID index 7. Node2 allocates local Prefix-SID label 16007 from the SRGB for prefix 1.1.1.7/32.

Note that Node2 learns that 1.1.1.7/32 is reachable via nexthop 1.1.1.4 (Node4). When Node2 receives an IP packet with destination 1.1.1.7/32, it pushes label 16007 on the packet and then pushes the Prefix-SID label of

Node4 (16004) on the packet to bring it to the BGP nexthop Node4. See Figure 6-4 (c). Since Node3 is the penultimate of Node4's Prefix Segment, it pops the label and forwards it to Node4. Node4 swaps label 16007 with label 16007 and forwards the packet to Node5. Node5 swaps label 16007 with label 16007, and finally, Node6 pops the label and forwards the packet to Node7. Example 6-9 shows the output of a traceroute command executed on Node2 towards Node7.

Example 6-9: Traceroute example on PE Node2 to PE Node7 – SR

```
RP/0/0/CPU0:xrvr-2#traceroute 1.1.1.7 source 1.1.1.2

Type escape sequence to abort.
Tracing the route to 1.1.1.7

 1  99.2.3.3 [MPLS: Labels 16004/16007 Exp 0] 9 msec  9 msec  9
msec
 2  99.3.4.4 [MPLS: Label 16007 Exp 0] 9 msec  9 msec  9 msec
 3  99.4.5.5 [MPLS: Label 16007 Exp 0] 9 msec  9 msec  9 msec
 4  99.5.6.6 [MPLS: Label 16007 Exp 0] 9 msec  9 msec  9 msec
 5  99.6.7.7 9 msec  9 msec  9 msec
```

The L3VPN service overlay network can be applied to the SR-enabled network. The difference with the regular BGP-LU situation is that the L3VPN traffic is now carried by Prefix Segments.

6.3 BGP Prefix-SID Advertisement

BGP Prefix Segments are specified in IETF [draft-ietf-idr-bgp-prefix-sid]. The BGP Prefix-SID is advertised in BGP as an optional, transitive BGP path attribute, which means that support for this attribute is not required and if the attribute is not recognized it should still be passed along to other peers. The (preliminary) attribute type is 40.

The BGP Prefix-SID attribute can be attached to prefixes from AFI/SAFI:

- Multiprotocol BGP labeled IPv4/IPv6 Unicast (IETF RFC 3107) for SR MPLS
- Multiprotocol BGP (IETF RFC 4760) unlabeled IPv6 Unicast for SRv6

The value field of the BGP Prefix-SID attribute contains one or more TLVs. The following TLVs have been defined:

- Label-Index TLV
- Originator SRGB TLV
- IPv6 SID TLV

The first two TLVs are only applicable to the SR MPLS data plane, while the third TLV only applies to SRv6, the SR IPv6 data plane. This book only covers the MPLS data plane for SR BGP.

The Label-index TLV is attached to an IPv4 or IPv6 labeled-unicast prefix and has the format shown in Figure 6-5.

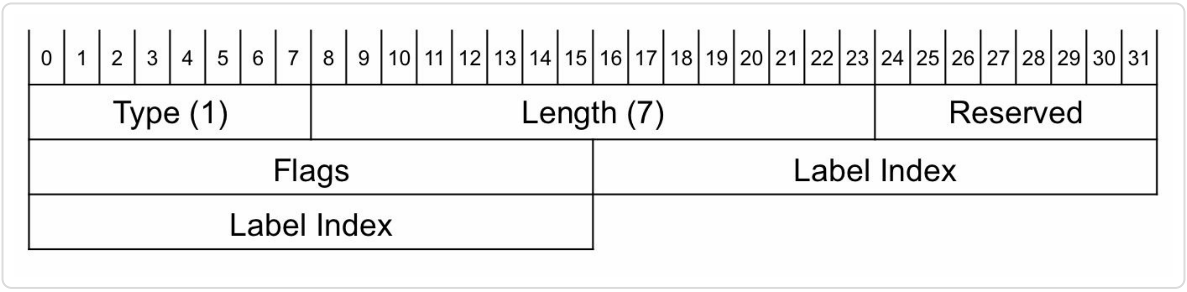


Figure 6-5: BGP Prefix-SID: Label-index TLV format

The fields in this TLV are:

- Flags: no flags have been defined yet
- Label Index: the Prefix-SID index, the label offset in the SRGB

The Originator SRGB TLV is an optional TLV that specifies the SRGB of the node that originates the prefix to which the BGP Prefix-SID attribute is attached. The Originator SRGB TLV has the format displayed in Figure 6-6.

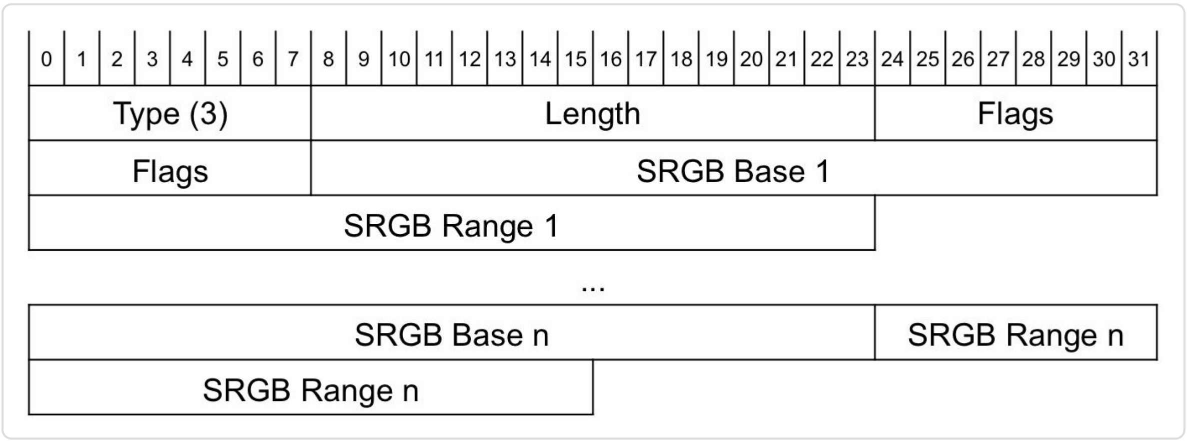


Figure 6-6: BGP Prefix-SID: Originator SRGB TLV format

The fields in this TLV are:

- Flags: no flags have been defined yet
- SRGB: consists of an SRGB Base (3 octets) and an SRGB Range (3

octets). Multiple SRGB fields can be present if the SRGB consists of multiple ranges.

The advertising node includes its allocated local label in the NLRI as per the classic BGP-LU operation. The originating node can advertise label value 3 (implicit-null) for a prefix to achieve PHP behavior. The receiving node programs the MPLS forwarding entries based on the received label as per classic BGP-LU behavior. A node that originates a prefix with BGP Prefix-SID includes the Prefix-SID index associated with the advertised prefix in the BGP Prefix-SID attribute Label Index TLV. The SR BGP node that receives the BGP update uses the SID index in the Label Index TLV as a *hint* to allocate a specific local label for the associated prefix; it allocates the label from the SRGB at the position indicated by the SID index. This local label is then advertised to the other BGP peers as per classic BGP-LU behavior. Also the transitive BGP Prefix-SID attribute is included unchanged in the prefix update.

6.4 Interoperability with non-SR BGP-LU

SR BGP automatically and seamlessly interoperates with non-SR nodes in the BGP-LU based network. The BGP Prefix-SID attribute is an optional transitive attribute, which means that nodes that do not understand the attribute just ignore it and forward it as is. The BGP Prefix-SID attribute contains a *hint* to allocate a label from the SRGB based on the Label Index TLV included in the attribute. If a node does not understand the attribute, it allocates a local label from the dynamic label range instead. This is the classic BGP-LU functionality. A node that supports SR BGP uses the hint to allocate the intended Prefix-SID local label from SRGB. The drawback is that in a mixed SR/non-SR network the Prefix-SID does not have a deterministic label on all nodes, therefore losing some of the Segment Routing benefits. Since the non-Segment Routing node re-advertises the unchanged BGP Prefix-SID attribute to its neighbors, all Segment Routing nodes in the network can use the Prefix-SID index as is originated. Note that at the time of writing this book, the Cisco IOS XR implementation only supported the addition of a Prefix-SID attribute to the BGP update only on the node that originates the prefix, which means that the originating node must be SR enabled.

Figure 6-7 illustrates the interoperability between SR BGP-LU and non-SR BGP-LU. It shows the same inter-AS topology as in the previous section, but in this case the nodes in AS1 do not support SR (BGP and IGP) while the nodes in AS2 have SR enabled. Since ASBR Node4 is a non-SR node, it does not understand the BGP Prefix-SID attribute that Node5 advertised with prefix 1.1.1.7/32. Therefore, Node4 does not allocate the Prefix-SID label, but a random label 90407 from the dynamic label range. It installs that label as incoming label in its forwarding table and advertises it with prefix 1.1.1.7/32 to Node2. The outgoing label is the label 16007 that it received from Node5. Since the Prefix-SID attribute is transitive, Node4 includes that attribute in its advertisement to Node2. This way, the other SR BGP nodes in the network could use that attribute to allocate the Prefix-SID label from their SRGB. In this example, Node2 does not support SR, so it will not use the Prefix-SID attribute.

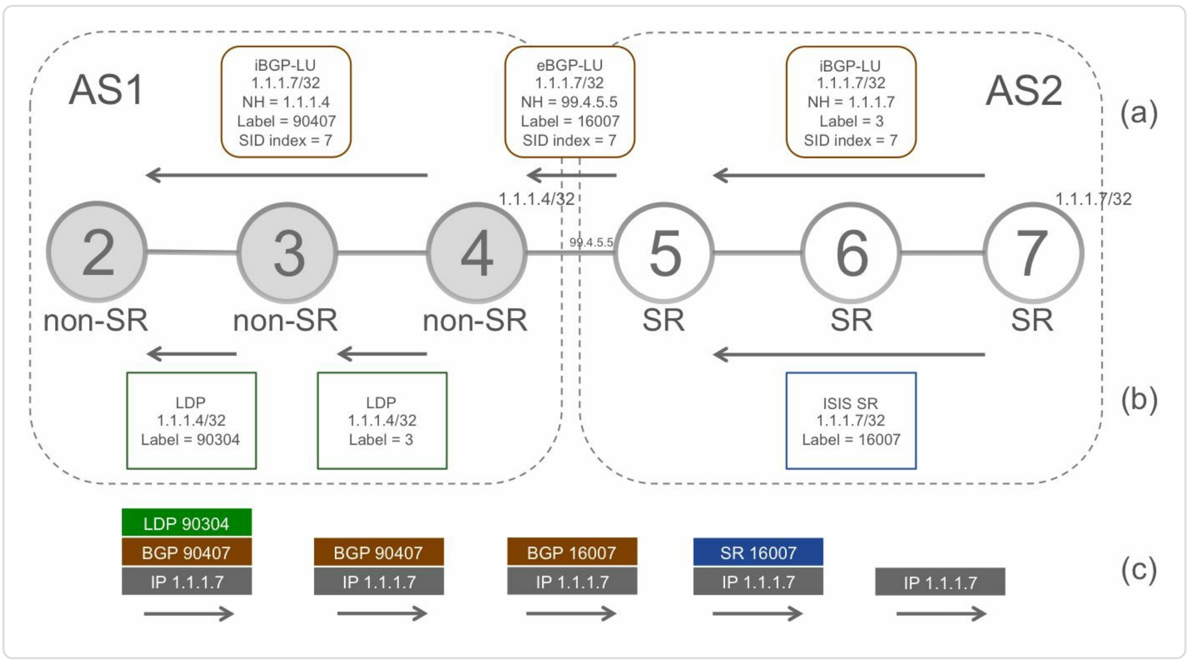

AS1
iBGP-LU 1.1.1.7/32 NH = 1.1.1.4 Label = 90407 SID index = 7
eBGP-LU 1.1.1.7/32 NH = 99.4.5.5 Label = 16007 SID index = 7
iBGP-LU 1.1.1.7/32 NH = 1.1.1.7 Label = 3 SID index = 7
AS2
(a)
1.1.1.4/32
99.4.5.5
1.1.1.7/32
2 non-SR
3 non-SR
4 non-SR
5 SR
6 SR
7 SR
LDP 1.1.1.4/32 Label = 90304
LDP 1.1.1.4/32 Label = 3
ISIS SR 1.1.1.7/32 Label = 16007
(b)
LDP 90304
BGP 90407
IP 1.1.1.7
BGP 90407
IP 1.1.1.7
BGP 16007
IP 1.1.1.7
SR 16007
IP 1.1.1.7
IP 1.1.1.7
(c)

Figure 6-7: Inter-AS Label switched path between PEs – SR/non-SR interworking

When an IP packet with destination 1.1.1.7 arrives on Node2, this node pushes two labels on the packet: the BGP label 90407 for 1.1.1.7/32 and the LDP label 90304 for the BGP nexthop prefix 1.1.1.4/32. It forwards the packet to Node, which pops the top label, since it is the penultimate hop towards Node4. Node4 swaps label 90407 with the Prefix-SID label 16007 and forwards the packet to Node5. Node5 swaps the label with the same Prefix-SID label 16007. Node6 then pops the label and forwards to Node7. Example 6-10 shows the output of a traceroute on Node2 towards Node7.

Example 6-10: Traceroute example on PE Node2 to PE Node7 – SR/non-SR interworking

```
RP/0/0/CPU0:xrvr-2#traceroute 1.1.1.7 source 1.1.1.2

Type escape sequence to abort.
Tracing the route to 1.1.1.7

 1  99.2.3.3 [MPLS: Labels 90304/90407 Exp 0] 19 msec  9 msec
9 msec
 2  99.3.4.4 [MPLS: Label 90407 Exp 0] 9 msec  9 msec  9 msec
 3  99.4.5.5 [MPLS: Label 16007 Exp 0] 9 msec  9 msec  9 msec
 4  99.5.6.6 [MPLS: Label 16007 Exp 0] 9 msec  9 msec  9 msec
 5  99.6.7.7 9 msec  9 msec  9 msec
```

HIGHLIGHT

BGP-LU has been extended to enable signaling of Prefix-SID and SRGB.

SR enabled BGP-LU implementations interoperate with non-SR BGP-LU implementations.

SR BGP Prefix-SIDs can be used over both EBGP and IBGP sessions and for various deployment designs like inter-AS (or inter-domain) MPLS end-to-end connectivity.

6.5 SR BGP in Seamless MPLS Architecture

The Seamless MPLS Architecture was designed to enable end-to-end MPLS connectivity for a large network comprising of multiple IGP domains. More details on this can be found in draft-ietf-mpls-seamless-mpls. Getting into the details of this architecture is beyond the scope of this book but both SR IGP and SR BGP with Prefix-SIDs fit into this design. In addition to providing basic MPLS connectivity, SR also enables using end-to-end TE for specific services across the multiple domain hierarchy, which will be covered in the next part of this book. As SR BGP extends and builds on BGP-LU, it can be enabled in this design for inter-domain signaling along with SR IGP doing the job of intra-domain signaling.

"In many cases, networks are organized as multiple IGP domains for scalability and/or administrative reasons (e.g. on the lines of regions or departments or network functionality). However, services need to be delivered end-to-end across them with specific SLAs. SR BGP extends the base SR IGP mechanisms to enable this end-to-end traffic engineering in a simple and scalable manner in these multi-domain architectures – something that was not possible or very complex to do until now. The ease of migration from existing network designs based on LDP/BGP-LU to SR IGP/BGP makes it possible to interwork and upgrade networks in a gradual manner."

— Ketan Talaulikar

6.6 BGP based Data Center Reminder

One of the earliest deployment use-cases for SR BGP has been the Data Center (DC) and we will cover this in some detail in the rest of this chapter. We first provide a quick reminder of how BGP based Data Center designs evolved.

6.6.1 From Classic DC to Clos Topology

Traditional Data Center networks have been designed to support conventional client-server services. The majority of the data flows in and out of such Data Center follow the traditional north-south traffic pattern. During the past decade the traditional three-tier hierarchical design, using Core, Aggregation, and Access layers, has been the predominant network architecture for Data Centers. It looks like a tree where the core functions as the trunk and with redundant uplinks and increasing bandwidth capacity moving upwards from server to Core. Services are concentrated, connected to the Aggregation Layer switches.

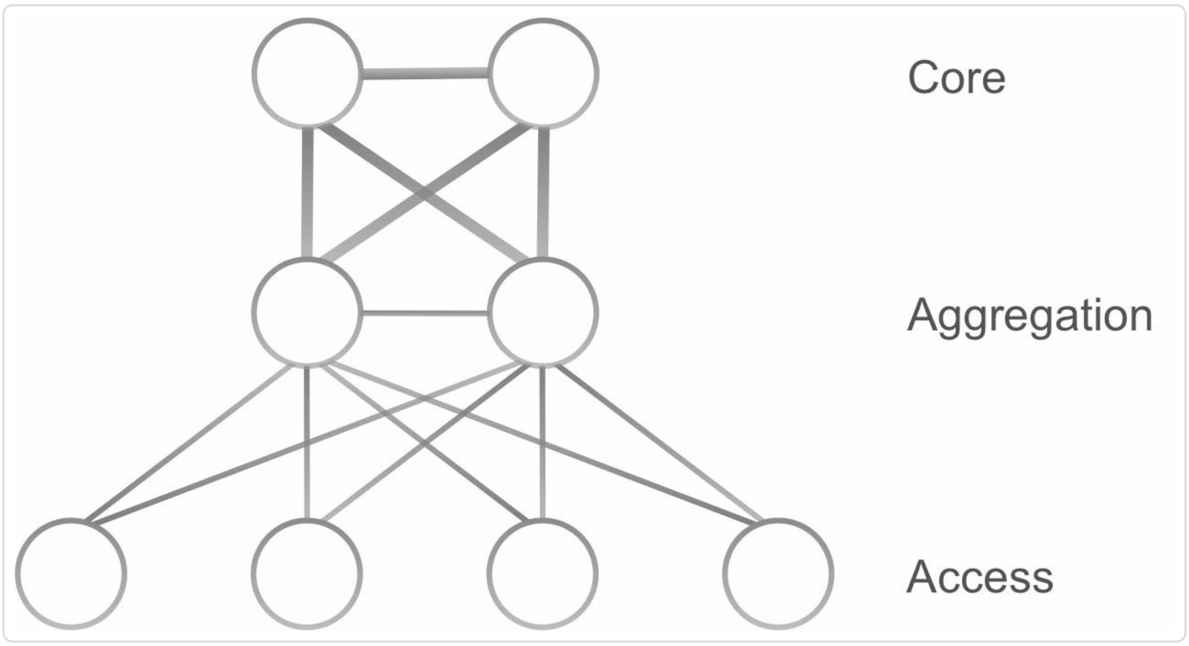


Figure 6-8: Traditional Data Center topology

Server virtualization and cloud-based services changed the requirements for the Data Center network design. Traditionally, the traffic flow was mainly north-south, from users to the applications hosted in the Data Center. With the new applications and services, the traffic patterns changed and the majority of the traffic nowadays remains within the Data Center, the east-west traffic or server-to-server traffic. This is caused by different factors such as the functional separation of application servers, storage, and databases that generate additional traffic traversing the Data Center for replication, backup, etc. Also, applications become more and more complex. To serve a single initial web query, a web server may need to collect data from a number of different applications and databases within the Data Center.

The classic three-tier Data Center was designed for north-south traffic, inter-application traffic was limited to the application racks. This design becomes less suitable for the requirements of the new server and application traffic requirements.

To overcome the limitations of the classic Data Center design, new and more scalable network architectures are used. One of the most common is the Clos Architecture. The name of this architecture comes from Charles Clos at Bell Laboratories who published a paper in 1953, “A Study of Non-Blocking Switching Networks”, where he describes a mathematical theory of a non-blocking, multi-stage network topology to switch telephone calls. Clos’s original telephone switch design ideas are now applied to the Data Center network topology. The Clos architecture provides high redundancy and multipath load-sharing with a simple design.

The Clos architecture is implemented as a multi-stage topology, using an odd number of stages, typically three. The middle stage is often called the Spine layer and the input and output stages are called the Leaf layers. Therefore this topology is often referred to as a "Spine and Leaf " network. If represented in the folded representation, where the input and output stages are folded together, the two layers of the topology become apparent. Figure 6-9 and Figure 6-10 illustrate a three-stage or two-layer Clos topology in its unfolded and folded representation respectively.

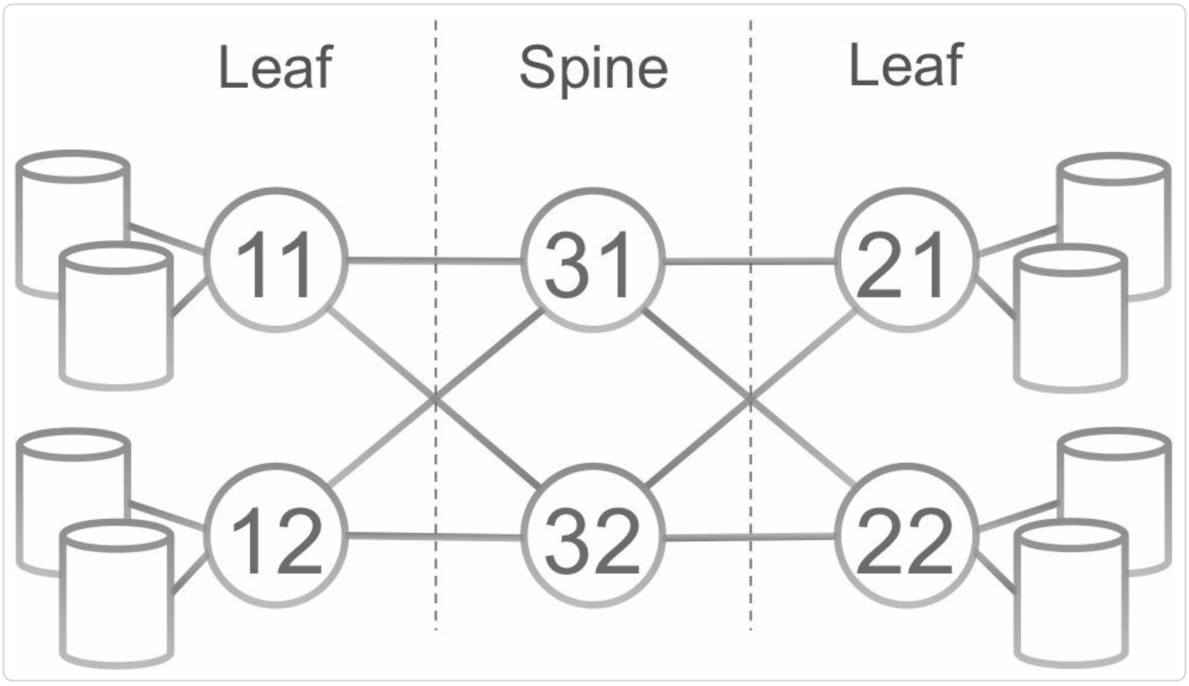


Figure 6-9: Clos topology – unfolded representation

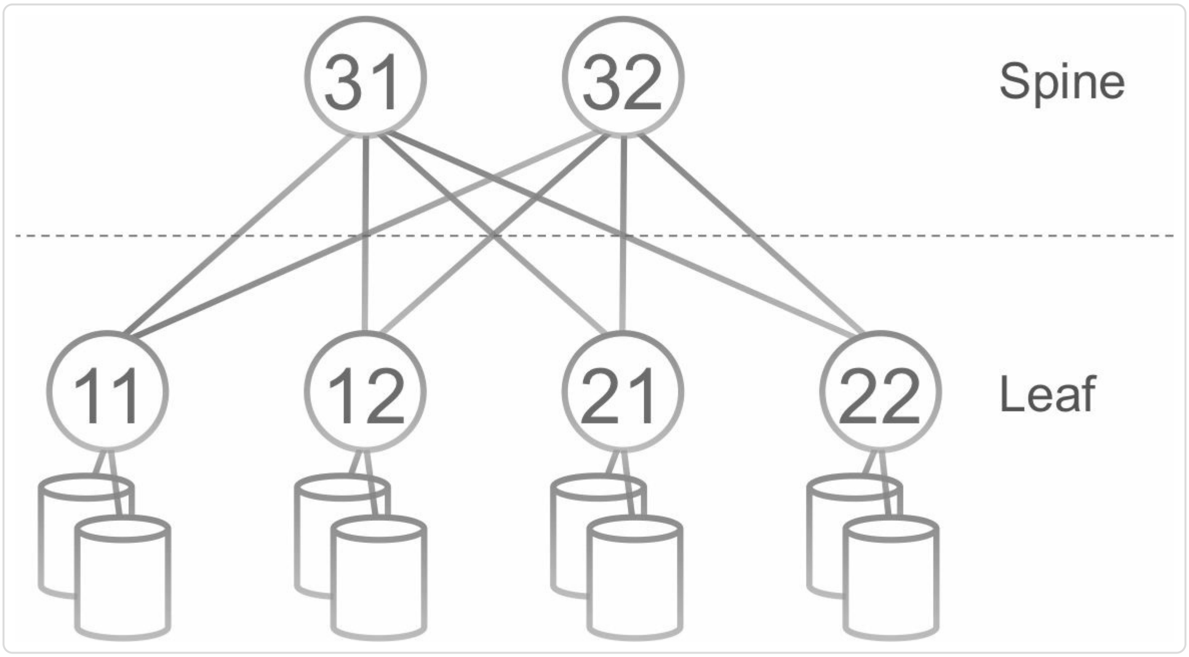


Figure 6-10: Clos topology – folded representation

The layers are also called “tiers”, with the tier number increasing moving from bottom to top, from Leaf to Spine. A Leaf node does not connect to other Leaf nodes but typically connects to all Spine nodes. This eliminates bandwidth aggregation since the bandwidth is the same at every layer. Spine nodes serve as a backbone interconnect for Leaf nodes, they do not connect to other Spine nodes.

The Spine and Leaf network makes up the Data Center “Fabric”. This Fabric provides high-bandwidth, low-latency, non-blocking[1] server-to-server connectivity. In general, servers are connected to the Leaf nodes that provide access to the Fabric. These leaf nodes that conventionally were placed at the top of the rack of server blades also come to be known as Top-Of-Rack or TOR nodes.

Scaling a Clos topology can be done by increasing the port density of the nodes or by adding more stages, like moving to a five-stage Clos topology as in Figure 6-11.

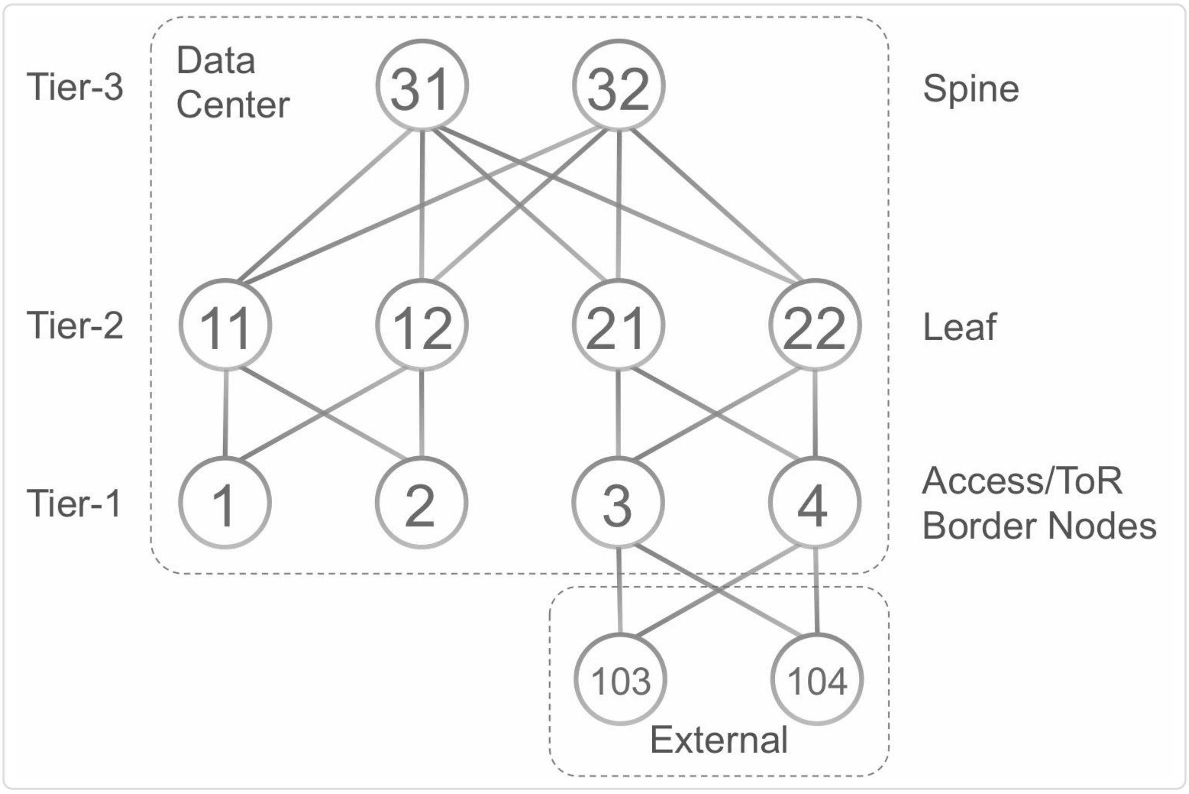


Figure 6-11: Five-stage Clos topology

6.6.2 BGP in Layer 3 Data Center

Traditionally Data Centers used a Layer-2 network, but due to its scaling and stability problems, many Data Centers moved to a Layer-2/Layer-3 hybrid solution or entirely Layer-3. Using a Layer-3 only design simplifies the network. This is the design that is assumed in this section.

Then there is the choice of the routing protocol. Very large Data Centers choose BGP as the only routing protocol ("BGP is the IGP"). Main reason to choose BGP is that the protocol is simple compared to the more complex link-state Interior Gateway Protocols. Simpler state machine and databases, "what you see is what you get": routes themselves are exchanged instead of link-state advertisements and no need to run Shortest Path First (SPF) computations. Also, BGP allows for some per hop traffic engineering. BGP is also more stable in the sense that the propagation of events is more limited than the domain-wide flooding using link-state protocols.

EBGP or IBGP

BGP can run in two modes, and each mode has a very different behavior when advertising routing information: External BGP (EBGP) and Internal BGP (IBGP). EBGP runs between peers in different Autonomous Systems (ASs). IBGP runs between peers in the same Autonomous Systems.

Although it uses the same protocol, there are a number of differences between the two flavors.

There is a difference in trust level that is assumed in the behavior: EBGP sessions where traditionally established between different organizations (Autonomous Systems), while IBGP sessions were used within the same organization.

EBGP sessions are typically established on a link that directly connects the BGP peers, while IBGP sessions are typically established between the loopback interfaces of remotely located BGP peers.

EBGP uses the AS-path attribute (listing all traversed ASs) as a loop prevention mechanism, while IBGP cannot use that mechanism since all peers are in the same AS. Therefore a number of route advertisements rules apply to prevent loops from occurring when IBGP is involved. Routes received from an EBGP peer are by default advertised to other EBGP and IBGP peers. Routes received from an IBGP peer are by default advertised to an EBGP peer but are by default not advertised to another IBGP peer. Due to these rules, IBGP requires a full mesh of sessions between the IBGP speakers in a network to ensure reachability for BGP prefixes throughout the network or use other mechanisms such as Route Reflectors or Confederations.

The natural properties of EBGP make it very simple and straightforward to use in the Data Center Fabric:

- Loop prevention using the AS-path attribute
- Default next-hop-self for advertised routes
- Automatic re-advertisement of routes in other EBGP sessions

IBGP can be used as well, making use of the following functionalities that make IBGP imitate the EBGP functionality:

- Route Reflectors with Originator-id and Cluster-list for loop prevention
- Enable next-hop-self for the re-advertised routes

IETF RFC 7938 describes how BGP can be used as the only routing protocol in a large scale Data Center. The topology in Figure 6-12 illustrates using BGP in a Data Center. Single-hop BGP sessions are used over direct links between the nodes, thus each link between nodes is a physical link as well as a BGP session. When using Internal BGP, then all nodes have the same Autonomous System (AS) number. When using External BGP, then the BGP neighbors have a different AS number. Private Use AS numbers from the range 64512-65534 are typically used to avoid conflicts with assigned AS numbers. However, for simplicity, the illustration uses shorter AS number that are outside of the Private Use range. In the illustration, a different AS number per node is indicated. This is one way of assigning AS numbers; other ways are possible.

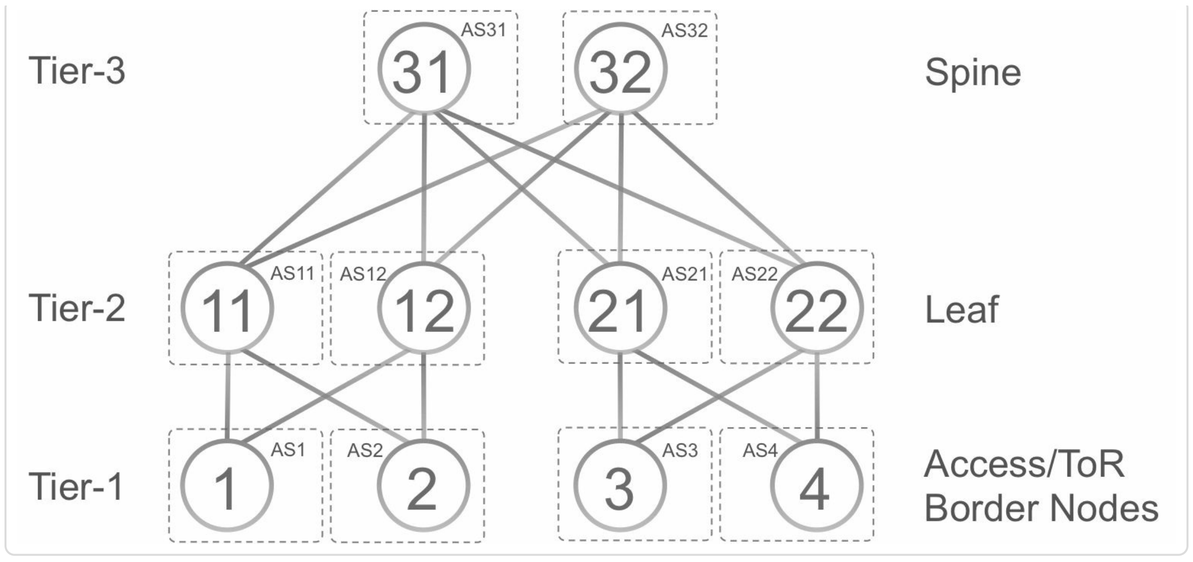


Figure 6-12: Example Data Center topology

6.6.3 MPLS in Data Center

Network overlays are virtual networks that share an underlay network, an underlying physical network. Network overlays can override the routing decisions made by the underlay network's routing protocols, often using a tunnel mechanism, without the need to modify the underlying network. Overlays also allow building large-scale networks using devices that only support small forwarding tables, since transit nodes in the underlay do not keep any state for overlay addresses, they simply forward overlay traffic to the underlay destination address in the outer packet header. Different overlay technologies exist, and one of them is MPLS.

6.6.4 BGP Labeled Unicast in Data Center

BGP-LU is illustrated using the example Data Center topology of Figure 6-12. This example only uses External BGP; the differences with Internal BGP are discussed later in this chapter.

To explain the functionality, only one of the equal cost paths between Node1 and Node3 is examined: the path

Node1→Node11→Node31→Node21→Node3. Figure 6-13 shows this path in its unfolded representation.

A single-hop EBGP Labeled Unicast (EBGP-LU) session is established over each link in the topology. Node1 peers with Node11, Node11 with Node31, and so on. Each node advertises its own loopback prefix in BGP. These prefixes will be used for the service overlay network. Strictly speaking, only the nodes participating in the service overlay must advertise their loopback prefix, only Node1 and Node3 in this example. However, these loopback prefixes will be used to carry the Prefix-SIDs when introducing Segment Routing in the Data Center Fabric, and SR can use each node's Prefix-SID for traffic steering.

Interface addresses are not advertised in this example. There is no real requirement to advertise these interface addresses in BGP since no service traffic is using interface addresses. A drawback is that it complicates operations and troubleshooting since these interface addresses are not pingable from remote nodes, and when doing a traceroute, then the non-reachable interface addresses show up in the traceroute output.

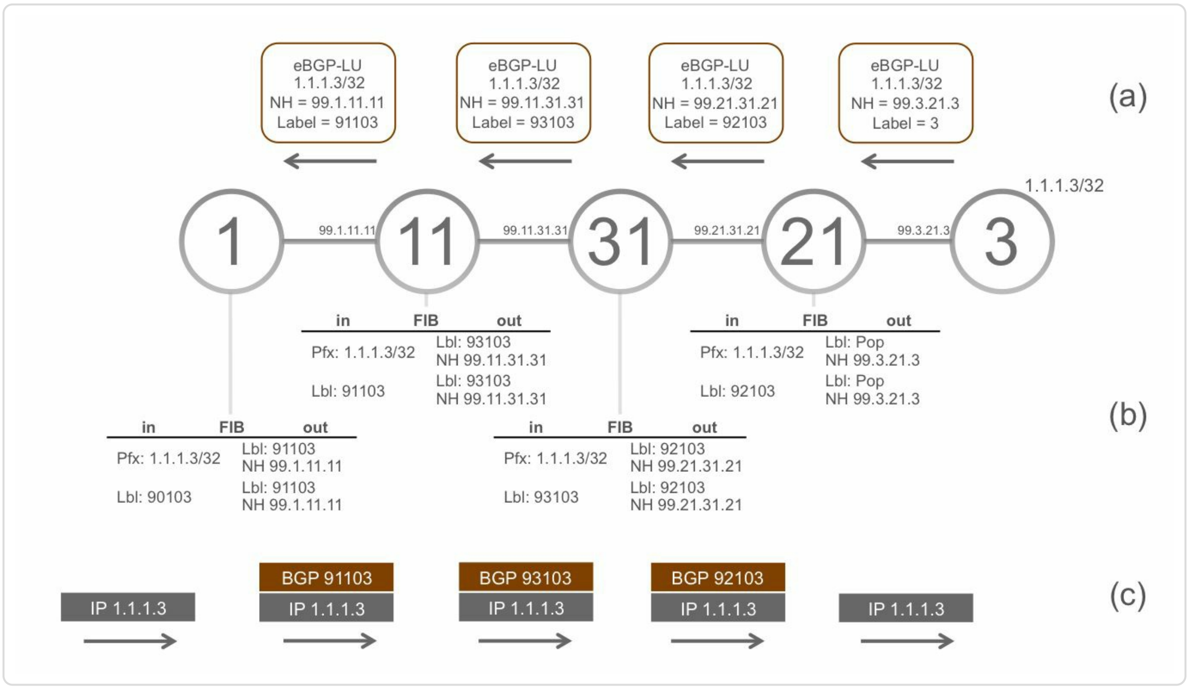


Figure 6-13: Unfolded path between Node1 and Node3

Node3 advertises its loopback prefix 1.1.1.3/32 to Node21 in EBGP-LU, with its interface address of the link to Node21 (99.3.21.3) as BGP nexthop (Figure 6-13 (a)). Since prefix 1.1.1.3/32 is local to Node3, Node3 advertises a label value 3 (implicit-null) with the prefix. After receiving the BGP-LU update, Node21 allocates a ("random") local label 92103 for this prefix from its dynamic label range. Node21 installs the label swap and label imposition forwarding entries for this prefix with outgoing label pop (Figure 6-13 (b)).

Node21 then advertises this prefix 1.1.1.3/32 with its local label 92103 to its EBGP-LU neighbor Node31. Node21 sets its interface address of the link to Node31 (99.21.31.21) as BGP nexthop since EBGP does "next-hop-self" by default (Figure 6-13 (a)). Node31 allocates a local label 93103 for this prefix and installs the label swap and imposition forwarding entries (Figure 6-13 (b)).

Similarly, Node31 and Node11 do the same operations and eventually Node1 learns the prefix 1.1.1.3/32, it allocates a local label 90103 and installs the forwarding entries for this prefix.

When Node1 receives an IP packet with destination 1.1.1.3, it pushes label 91103 on the packet and forwards it to Node11. Node11 swaps label 91103 with label 93103 and forwards the packet to Node31. Node31 swaps label 93103 with label 92103, and finally, Node21 pops the label and forwards the packet to Node3. This is illustrated in Figure 6-13 (c).

BGP-LU provides the MPLS connectivity in the Data Center Fabric, the "underlay" network. This MPLS "underlay" can then be used by service overlay networks. In this example, an L3VPN overlay is introduced to enable L3VPN connectivity between devices (such as servers or Virtual Machines (VMs)) connected to Node1 and Node3. For this purpose, a Service Route Reflector is introduced in the network: Node41. Node41 is connected to the Tier-3 node in this example. In this example, Node1 and Node3 are physical nodes participating in the service overlay. The service overlay functionality could be moved to virtual forwarders or servers.

In this example, a multi-hop EBGP session is established between Node1, Node3, and the Route Reflector (RR) Node41. L3VPN vpnv4 routes are exchanged over this EBGP session. Node41 is then strictly speaking not a Route Reflector since Route Reflectors are an IBGP functionality, but EBGP can also "reflect" routes. See illustration in Figure 6-14.

If Internal BGP is preferred for the service overlay routes, it is possible to enable a second BGP instance on Node1 and Node3 (multi-instance BGP). This second BGP instance can be in the same AS as the RR, allowing IBGP sessions to this RR.

Server19 in Figure 6-14 is connected to Node1 and Server39 Node3. The interfaces to the servers are configured in a vrf named *SVC*. Node3 allocates a local VPN label 90039 for vrf *SVC*. Node3 installs a *Pop-and-lookup* label entry for the VPN label; the VPN label of the incoming packet is popped and the destination IP address of the packet is then looked up in the vrf *SVC* forwarding table. Node3 then advertises the prefix 99.3.39.0/24 of the vrf interface to Server39 with the VPN label 90039 to the Route Reflector Node41. Node41 then re-advertises this prefix to Node1. Node41 is configured to not update the BGP nexthop attribute in the prefix update (`next-hop-unchanged`). Node1 receives the prefix update and installs the forwarding entry in the vrf forwarding table.

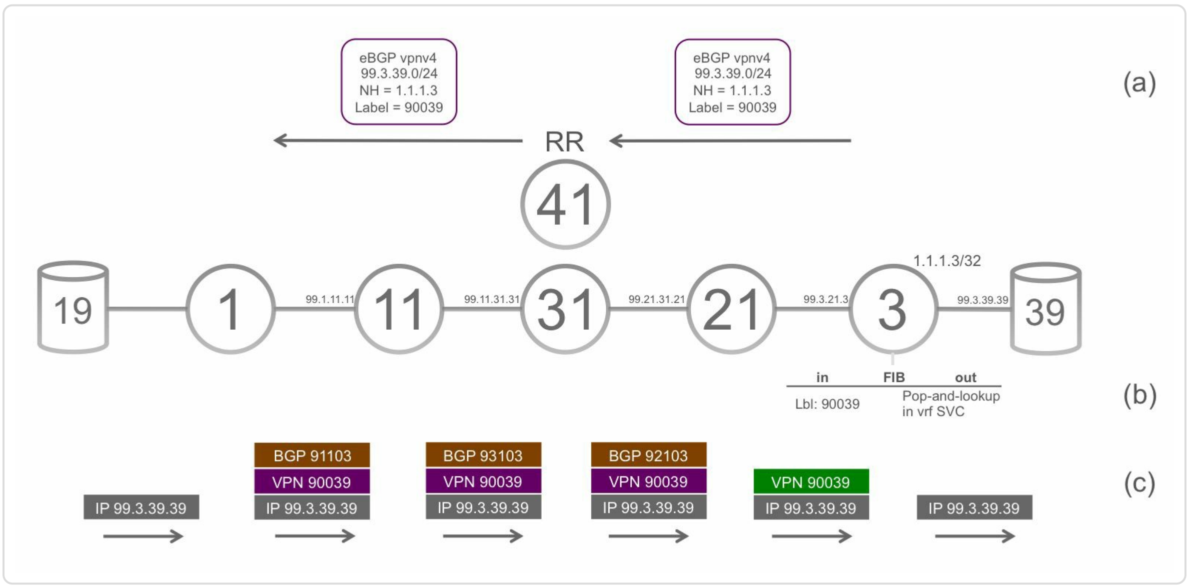


Figure 6-14: Unfolded path with service overlay

When Node1 receives an IP packet with destination 99.3.39.39 on the vrf *SVC* interface, it imposes two labels on the packet: the VPN label 90039 received from Node3 and the BGP-LU label 91103 for the BGP nexthop prefix 1.1.1.3/32. The packet then follows the path with the label stack illustrated in Figure 6-14 (c). Eventually, the packet arrives at Node3 with top label 90039. Node3 does a pop-and-lookup and forwards the packet to Server39 with address 99.3.39.39.

In the example, only a single path from Node1 to Node3 was examined, but ECMP is abundantly available in a Data Center Fabric. To enable using the available ECMP, the BGP multi-path functionality is used. By default, BGP just installs the forwarding entry for the BGP best path to a prefix. With BGP multi-path, BGP installs a selection of other available paths to the prefix in the forwarding table, on top of the BGP best path. In the topology of Figure 6-12, there are eight possible paths to go from Node1 to Node3. Note that Node1 only has two forwarding entries towards Node3 in its forwarding table, corresponding to its two upstream neighbors. These two paths fan out to eight paths on the following hops.

A server can be multi-homed to more than one Fabric node, in which case an equivalent mechanism can be used in the service overlay network to load-balance over the multi-homed connections. This introduces a second layer of load-balancing. If for example, a server is multi-homed to Node3 and Node4 (see Figure 6-15) then Node1 load-balances traffic to that server over the paths to Node3 and Node4. This is the first layer of ECMP. The paths to Node3 and Node4 use the available ECMP of the Data Center Fabric. This is the second layer of ECMP.

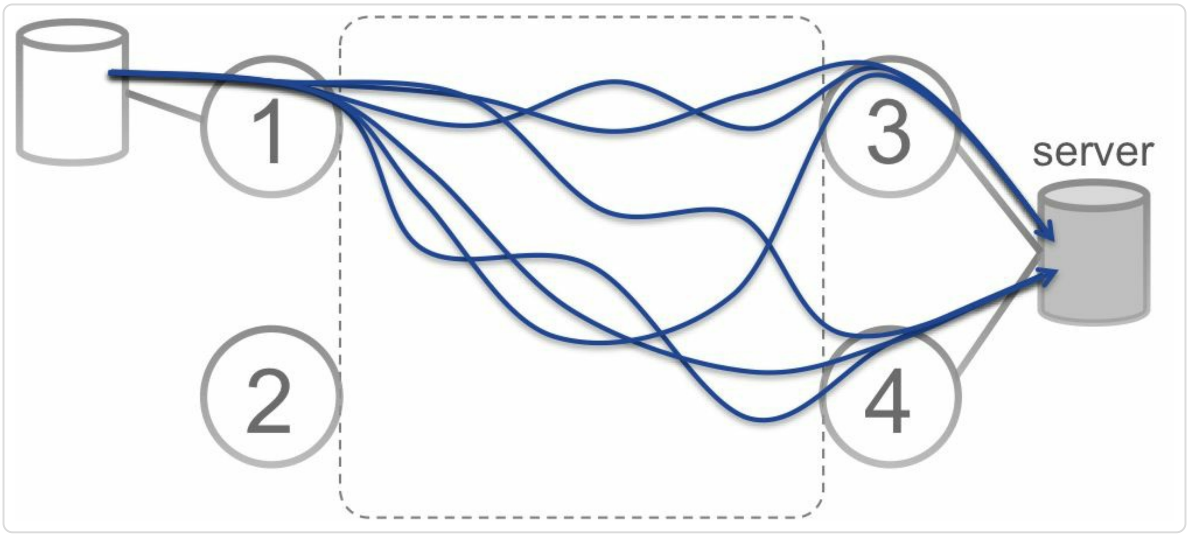


Figure 6-15: Traffic is load-balanced on the service overlay and in the DC Fabric

6.7 SR Benefits in DC

A number of open problems still exist in modern Data Center networks. Segment Routing can solve these problems. The SR control plane and data plane are the foundation of the solution, but additional enhancements are required to implement the proposed solutions, such as a centralized controller and an SR-supporting host networking stack.

6.7.1 Load-Balancing Efficiency

Traditional load-balancing over multiple (equal cost) paths is mostly realized per traffic flow; packets are steered on a path based on the hash computation over a number of elements in the packet header. This results in packets with common header fields (like packets of the same traffic flow) being steered on the same path through the network. Once a traffic flow is placed on a path, it stays on that path until the traffic flow stops. The advantage of per-flow load-balancing is that it prevents packet reordering since all packets within a flow follow the same path through the network.

The load-balancing accuracy of this mechanism depends on the characteristics of the traffic flows: the best efficiency is achieved for short-lived flows; the efficiency of per-flow load-balancing decreases when there are many long-lived traffic flows. Even if the flows are perfectly balanced over the available paths, traffic is not due to the sizes and rates of the flows. These traffic flow sizes and rates are heavy tailed; a few traffic flows contribute most of the traffic. Large, long-lived *elephant* flows are likely to affect the performance of smaller, short-lived *mouse* flows.

Inaccurate traffic balancing potentially leads to worse network performance.

Per-packet load-balancing provides very accurate traffic load-balancing, no traffic load imbalances occur with such scheme. A drawback is that per-packet load-balancing causes packet re-ordering, which is undesirable.

Flowlets [FLOWLET] combine the accuracy of per-packet load-balancing with the property of per-flow load-balancing that prevents out-of-order packets. Flowlets are basically bursts of packets belonging to a larger traffic flow. A host or ToR may split a traffic flow into flowlets and select different Segment Routing instructions per flowlet, and route them over different paths. Since flowlets are short-lived, the load-balancing efficiency increases, while maintaining packet order within each flowlet.

This is illustrated in Figure 6-16. The traffic flows labeled 1 and 3 are *elephant* flows; the traffic flow labeled 2 is a *mouse* flow. The load-balancing happens to steer the traffic as indicated in the illustration Figure 6-16 (a), causing a highly utilized link between Node21 and Node3. A controller can avoid the hotspots by placing some of the traffic flows on other paths, see Figure 6-16 (b). The host can divide the traffic flows in flowlets and steer the flowlets on different paths through the network by imposing different Segment Lists on the flowlets, see Figure 6-16 (c).

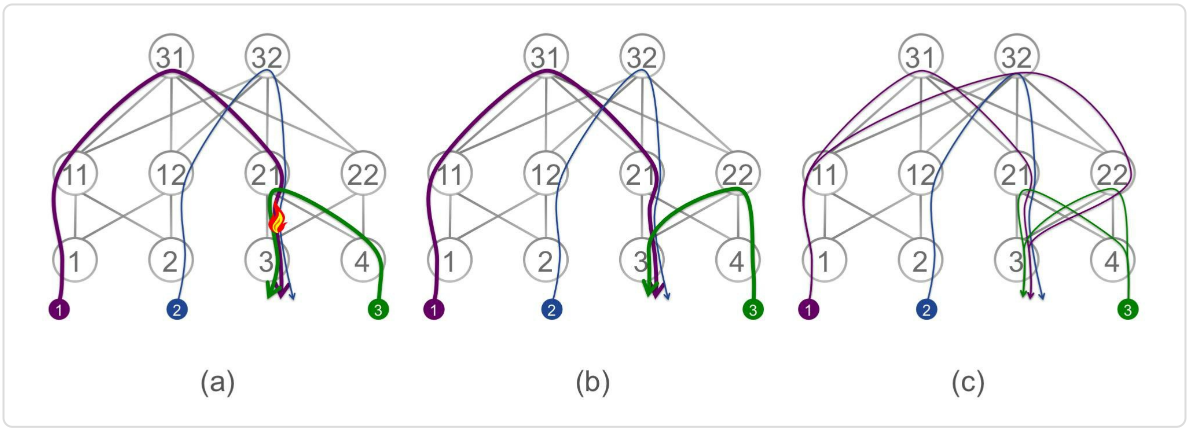


Figure 6-16: Load-balancing efficiency example

6.7.2 Non-Oblivious Routing

The load-balancing mechanism is not aware of any network imbalances. If the symmetry of the network is broken, for example due to a link failure, then utilization hotspots may appear. Remote nodes are not aware that one or some members of the ECMP set are unavailable and keep sending the same amount of traffic on each path, which now has to be distributed over fewer links.

This phenomenon is illustrated in Figure 6-17. The link between Node21 and Node31 failed but the other nodes are not aware of this failure and keep load-balancing traffic as if all links are available. This leads to a utilization hotspot on the link between Node31 and Node22, see Figure 6-17 (a). A controller detects the traffic imbalance and steers the flow with label 2 over another path for the duration of the failure, see Figure 6-17 (b).

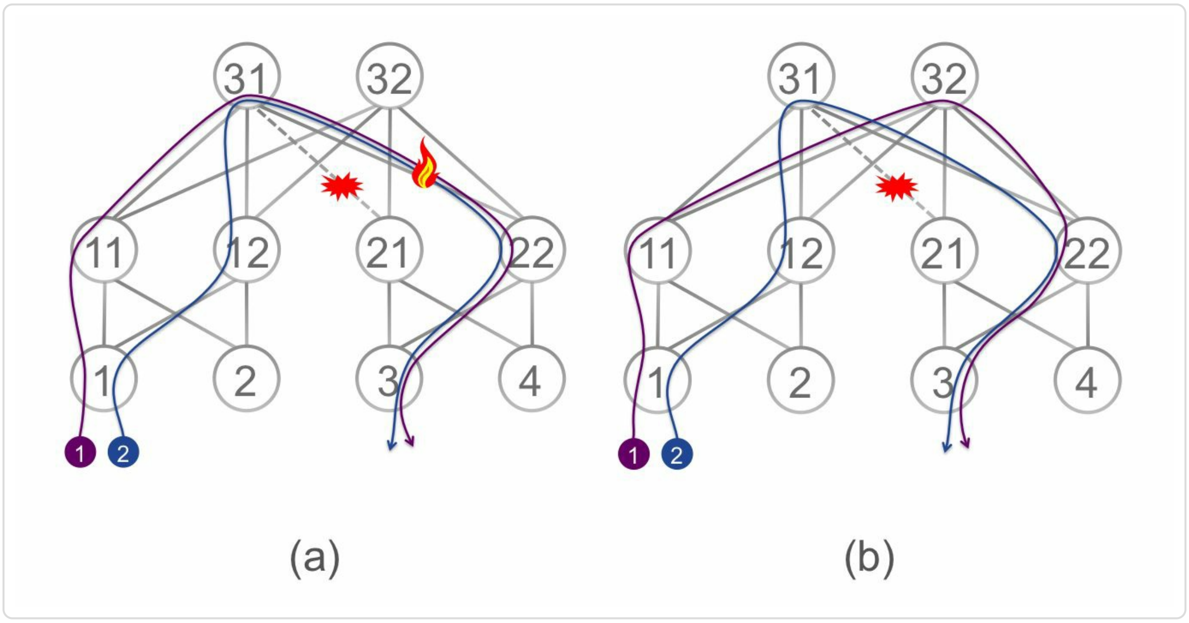


Figure 6-17: Non-oblivious routing example

6.7.3 Performance-Aware Routing

Hosts have no visibility of the path the traffic flows follow through the network, the network is a "black-box". Any traffic flow can follow a different path. This makes it for instance impossible to take the performance of a path into account when starting a new connection. Performance-aware routing can solve thus issue; a host can steer traffic based on characteristics of the path(s), derived by itself or from controller, and steer traffic away from less performing paths.

6.7.4 Deterministic Network Probing

Isolating faults in a network with multiple parallel paths and ECMP-based routing is non-trivial due to lack of determinism. Each new connection between two hosts may take a different path. This makes troubleshooting and failure reproduction more difficult. SR makes it possible to steer probe packet on exact paths through the network. A probing agent can thus send actively send probes through the network and isolate faulty nodes or links by correlating the probing results. Also see draft-ietf-spring-sr-oam-requirement and draft-ietf-spring-oam-usecase.

HIGHLIGHT: BGP SR in Data Center

BGP Prefix SID brings improved Traffic Engineering capabilities in the DC.

The SR traffic steering capabilities can be utilized to solve existing problems in the DC.

Nodes in the DC can steer traffic (as flows or flowlets) to improve load-balancing efficiency, avoid hotspots, or use the best performing path.

SR OAM probes can be steered on any path through the network to validate path liveliness.

6.8 BGP Prefix-SID in Data Center

We have seen previously in sections 6.2 ("BGP Prefix-SID") and 6.3 ("BGP Prefix-SID Advertisement") how SR BGP extends the BGP-LU mechanism to also signal the Prefix-SID for BGP. Here we will look at an example of its application for enabling SR BGP in a Data Center.

In the Data Center topology of Figure 6-12, a single path from Node1 to Node3 is now examined, the path Node1→Node11→Node31→Node21→Node3. The path is represented as an unfolded path in Figure 6-18. Same as for the BGP-LU example used earlier in this chapter, External BGP sessions are established between the nodes. All nodes in the network have SR BGP enabled, using the same SRGB [16000-23999]. A Prefix-SID with SID index 3 is configured on Node3 for its loopback prefix 1.1.1.3/32. Node3 advertises prefix 1.1.1.3/32 to Node21 in BGP-LU, with label value 3 (implicit-null). The BGP nexthop for this prefix is the interface address 99.21.3.3. Node3 also adds the BGP Prefix-SID attribute to the update message, specifying Prefix-SID index 3. Notice that both a label value and a SID index are advertised with the prefix (Figure 6-18 (a)).

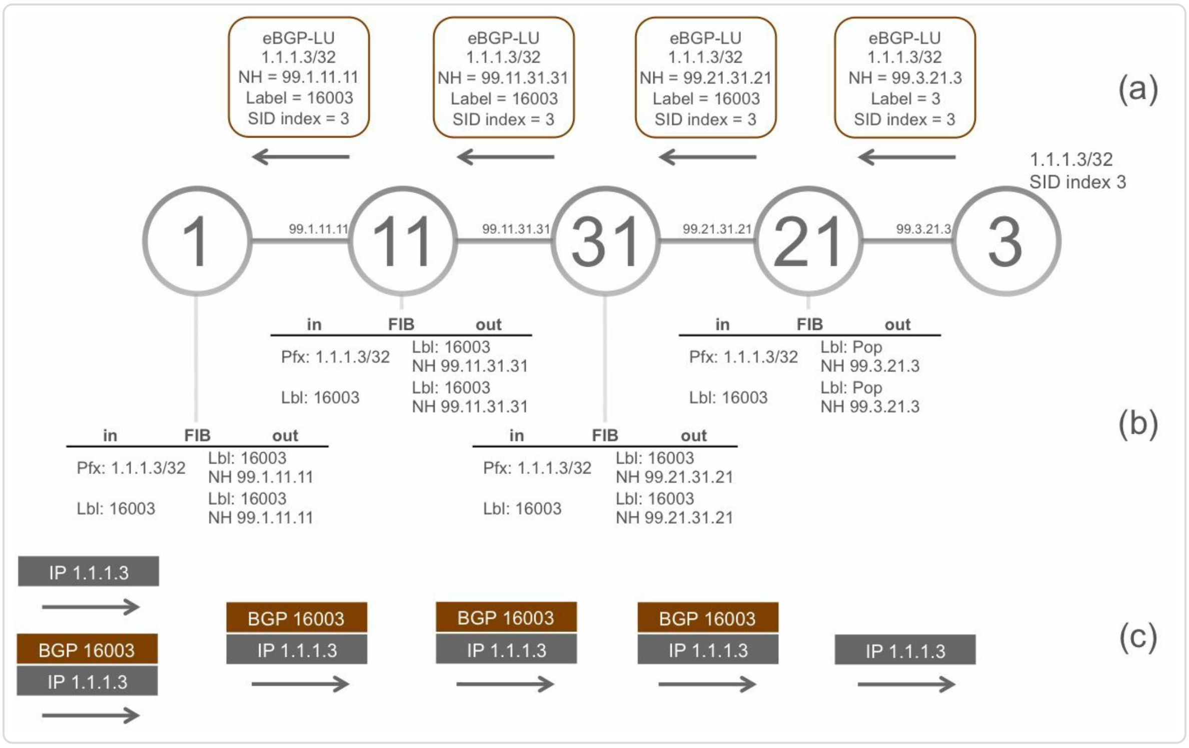


Figure 6-18: SR BGP: unfolded path from Node1 to Node3

When Node21 receives the prefix update, it allocates a local label for prefix 1.1.1.3/32. This local label is not allocated from the dynamic label range as with regular BGP-LU, but it is allocated from the SRGB, based on the Prefix-SID index attribute. Node21 allocates local label 16003 (= 16000 + 3) for this prefix. Node21 installs the MPLS imposition and swap forwarding entries for prefix 1.1.1.3/32 with outgoing label pop (Figure 6-18 (b)) and then re-advertises the prefix with label 16003 and SID index 3 to its EBGP-LU neighbor Node31. The BGP Prefix-SID attribute is re-advertised unchanged with the prefix. Node21 changes the BGP nexthop for this prefix to its own BGP session address 99.31.21.21 (default EBGP "next-hop-self" behavior).

Node31 and Node11 do the same operations and eventually Node1 learns the prefix 1.1.1.3/32 with label 16003 and Prefix-SID index 3. Node1 allocates local Prefix-SID label 16003 from the SRGB for prefix 1.1.1.3/32.

When Node1 receives an IP packet with destination 1.1.1.3/32, it pushes label 16003 on the packet and forwards it to Node11. See Figure 6-18 (b). Node11 swaps label 16003 with label 16003 and forwards the packet to Node31. Node31 swaps label 16003 with label 16003, and finally, Node21 pops the label and forwards the packet to Node3. When an MPLS-capable device that is connected to Node1 sends an MPLS packet with top label 16003 to Node1, then Node1 forwards this packet along the Prefix Segment to Node3.

The recommendation is to use the same SRGB on all devices in the network. This way the same label value is used for a Prefix-SID throughout the network. This deterministic label value largely simplifies operations and troubleshooting of the network and eases programming traffic steering policies.

The service overlay network can be applied to the SR-enabled Data Center network. The difference with the regular BGP-LU situation is that the overlay traffic is now carried by Prefix Segments.

There is no change in BGP multi-path functionality when using BGP Prefix-SIDs. BGP Prefix-SIDs are ECMP-aware, they naturally make use of all available BGP multi-paths.

SR BGP Anycast-SID

Also Anycast-SIDs can be used with SR BGP. Anycast-SIDs are Prefix-SIDs advertised by multiple nodes. For example, if the two Tier-1 nodes Node3 and Node4 in Figure 6-11 are peering nodes, connecting to the outside world, and they advertise the same Anycast-SID, then traffic steered to the Anycast-SID can be automatically load-balanced between the two nodes. Or if a subset of the Tier-3 nodes advertises an Anycast-SID then traffic can be steered via that subset by using the Anycast-SID. Using Anycast-SIDs also brings node redundancy; if one node in an anycast set fails then traffic will deviate to the other nodes in the set.

Previously, we have discussed the aspects of interoperability with non-SR BGP-LU implementations in section 6.4, "Interoperability with non-SR BGP-LU") and now we look at the same concept in our Data Center example.

Figure 6-19 illustrates the interoperability between SR BGP-LU and non-SR BGP-LU. It shows an unfolded path from Node1 to Node3, but Node21 in the path does not support SR BGP. This node does not understand the BGP Prefix-SID attribute that Node3 advertised with prefix 1.1.1.3/32. Therefore, Node21 does not allocate the Prefix-SID label but a random label 92103 from the dynamic label range. It installs that label as incoming label in its forwarding table and advertises it with prefix 1.1.1.3/32 to Node31. Since the Prefix-SID attribute is transitive, Node21 includes that attribute in its advertisement to Node31. This way, the other SR BGP nodes in the network can use that attribute to allocate the Prefix-SID label from their SRGB. The label imposed on a packet destined for 1.1.1.3 does not stay the same in this topology. Notice that the Prefix-SID label 16003 changes to the random label 92103 when Node31 forwards the packet to the non-SR BGP Node21.

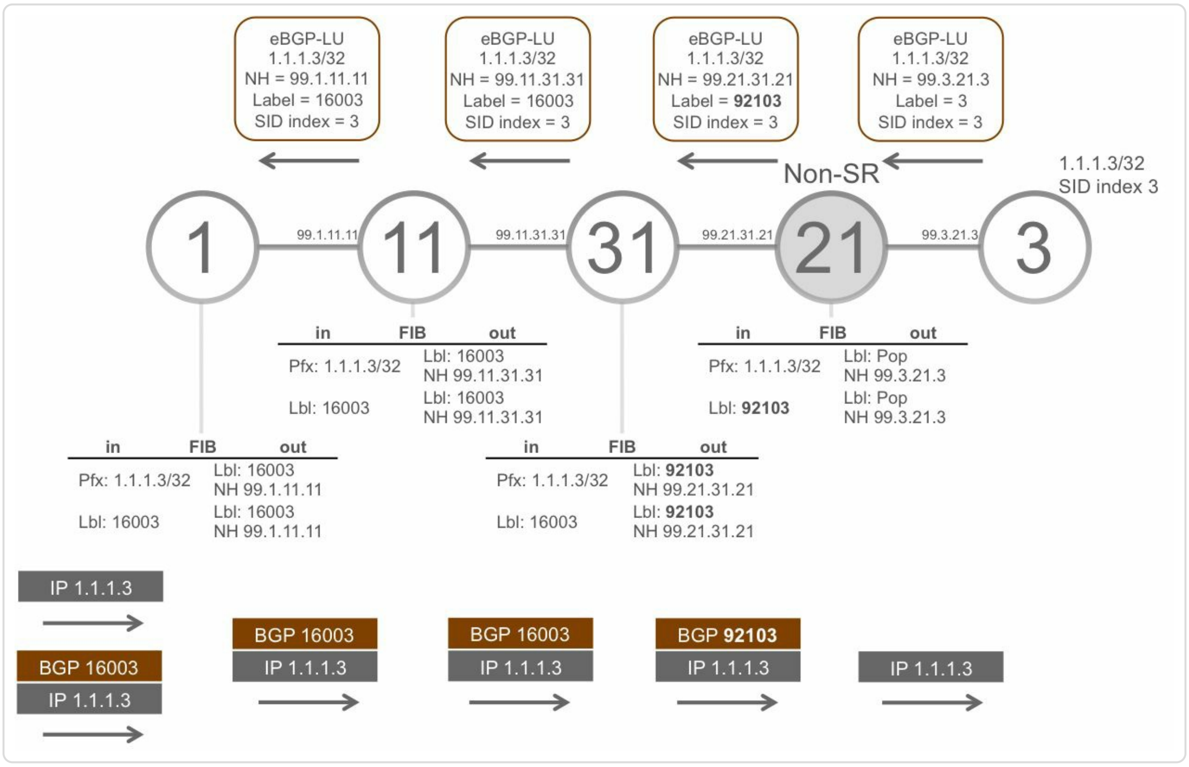


Figure 6-19: Interoperability between SR and non-SR BGP-LU

6.8.1 Configuring E-BGP Prefix Segments

In this section, the Clos network topology of Figure 6-12 is initially reduced to the topology of Figure 6-20 by removing the other nodes. This simplifies the configuration and the behavior. The reduced topology has two Tier-1 nodes (Node1 and Node3), two Tier-2 nodes (Node11 and Node12) and one Tier-3 node (Node31). Later in this section, the topology is expanded to its original size.

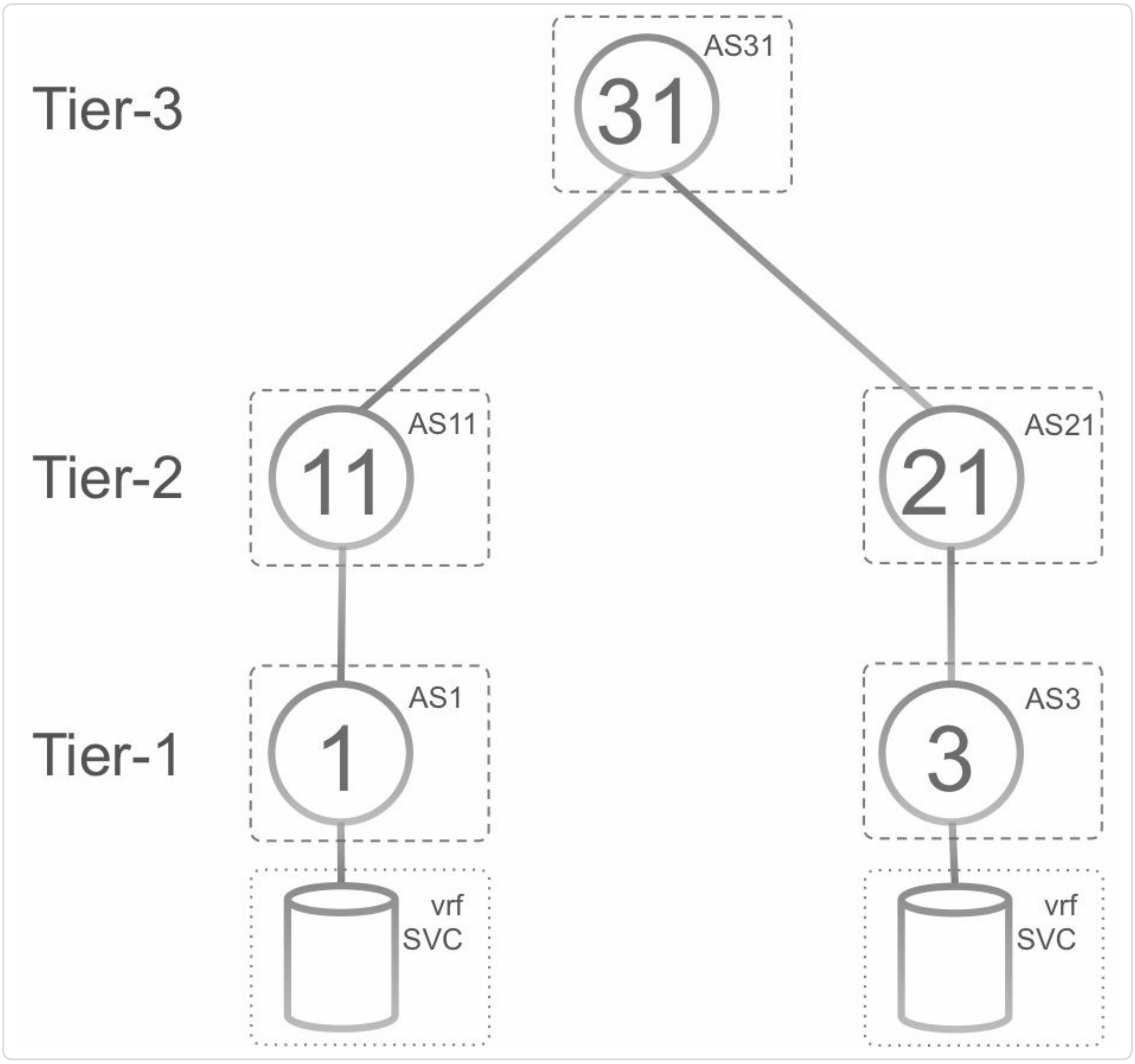


Figure 6-20: Reduced example Data Center topology

Each node has a different AS number; with the AS number equal to the number of the node. Each node establishes single-hop EBGP-LU sessions with its connected neighbors. SR BGP is configured on all nodes, providing SR MPLS transport in the Data Center. An L3VPN service is deployed between the Tier-1 nodes Node1 and Node3. These Tier-1 nodes offer a vrf interface to VMs connected to them (“vrf SVC”).

The BGP configuration of the Tier-1 Node3 is shown in Example 6-11.

Example 6-11: EBGP configuration of Node3

```
interface Loopback0
 ipv4 address 1.1.1.3 255.255.255.255
!
route-policy SID($SID)
  set label-index $SID
end-policy
!
route-policy bgp_in
  pass
end-policy
!
route-policy bgp_out
  pass
end-policy
!
router bgp 3
 bgp router-id 1.1.1.3
 address-family ipv4 unicast
  network 1.1.1.3/32 route-policy SID(3)
  allocate-label all
 !
 neighbor-group TIER2
  address-family ipv4 labeled-unicast
   route-policy bgp_in in
   route-policy bgp_out out
  !
 !
 neighbor 99.3.21.21
  remote-as 21
  use neighbor-group TIER2
  description eBGP peer xrvr-21
 !
!
router static
 address-family ipv4 unicast
  99.3.21.21/32 GigabitEthernet0/0/0/0
 !
!
segment-routing
 global-block 16000 23999
!
```

The BGP configuration starts on line 16 of the output. Node3 is in AS number 3. IPv4 unicast address-family is enabled and labels are allocated for all prefixes originated by this node in the address-family IPv4 unicast (`allocate-label all` on line 20).

The loopback prefix is advertised in BGP by using a `network` configuration under address-family ipv4 unicast (line 19). A route-policy `SID(3)` is applied to the network statement of prefix 1.1.1.3/32. This configuration sets the value of the Label-index TLV in the BGP Prefix-SID attribute for prefix 1.1.1.3/32 to value 3.

Lines 4 to 6 in the output show the definition of the route-policy. It uses the Route-Policy Language (RPL) statement `set label-index <n>` to set the Prefix-SID index. Note that this route-policy uses an argument `$SID`. This makes it possible to specify the Prefix-SID index value when attaching the route-policy to the network statement, as in line 19 where `SID(3)` is configured. This is only an example; other ways of setting the label-index in RPL are possible, such as using a prefix-set.

A neighbor-group `TIER2` is defined in lines 22 to 25 to simplify the configuration of the BGP neighbors. A neighbor-group is used as a template and applied using the `use neighbor-group` configuration under the neighbor. Address-family IPv4 labeled-unicast is enabled under this neighbor-group. Using ingress and egress route-policies is mandatory by default for EBGP sessions. In this example, they permit all prefixes (`pass`). The neighbor-group TIER2 is applied to neighbor 99.3.21.21 (line 30), which is Tier-2 Node21.

A static route is configured for the BGP neighbor’s interface host prefix 99.3.21.21/32, see line 36. Such static route is required for all directly connected BGP-LU neighbors in order to resolve the BGP-LU routes.

MPLS forwarding is automatically enabled on interfaces to directly connected EBGP-LU neighbors.

The Segment Routing Global Block (SRGB) is globally configured as the label range [16000-23999], see lines 39 and 40.

A packet capture of the BGP Update message from Node3 to Node21 is displayed in Example 6-12. The MP_REACH_NLRI attribute that contains the reachable destinations together with their next hop is displayed in lines 8 to 25. The update is for Address-family IPv4 Labeled Unicast; the BGP nexthop is 99.3.21.3. The NLRI contains the prefix 1.1.1.3/32 and the label value 3 (implicit-null).

The update message also contains a Prefix-SID attribute, on lines 57 to 69. The Label-Index TLV that is included contains the SID index 3. And the update message contains the Origin attribute IGP on line 26; the AS-path attribute on line 35 contains only Node3's AS 3; the Multi Exit Discriminator (MED) attribute 0 on line 48.

Example 6-12: BGP-LU update message sent by Node3

```
 1 Border Gateway Protocol - UPDATE Message
 2     Marker: ffffffffffffffffffffffffffffffff
 3     Length: 77
 4     Type: UPDATE Message (2)
 5     Unfeasible routes length: 0 bytes
 6     Total path attribute length: 54 bytes
 7     Path attributes
 8         MP_REACH_NLRI (21 bytes)
 9             Flags: 0x90 (Optional, Non-transitive, Complete,
Extended Length)
10                 1... .... = Optional: Optional
11                 .0.. .... = Transitive: Non-transitive
12                 ..0. .... = Partial: Complete
```

```
                ...1 .... = Length: Extended length
            Type code: MP_REACH_NLRI (14)
            Length: 17 bytes
            Address family: IPv4 (1)
            Subsequent address family identifier: Labeled Unicast (4)
            Next hop network address (4 bytes)
                Next hop: 99.3.21.3 (4)
            Subnetwork points of attachment: 0
            Network layer reachability information (8 bytes)
                Label Stack=3 (bottom) IPv4=1.1.1.3/32
                    MP Reach NLRI Prefix length: 56
                    MP Reach NLRI Label Stack: 3 (bottom)
                    MP Reach NLRI IPv4 prefix: 1.1.1.3 (1.1.1.3)
        ORIGIN: IGP (4 bytes)
            Flags: 0x40 (Well-known, Transitive, Complete)
                0... .... = Optional: Well-known
                .1.. .... = Transitive: Transitive
                ..0. .... = Partial: Complete
                ...0 .... = Length: Regular length
            Type code: ORIGIN (1)
            Length: 1 byte
            Origin: IGP (0)
        AS_PATH: 3 (9 bytes)
            Flags: 0x40 (Well-known, Transitive, Complete)
                0... .... = Optional: Well-known
                .1.. .... = Transitive: Transitive
                ..0. .... = Partial: Complete
                ...0 .... = Length: Regular length
            Type code: AS_PATH (2)
            Length: 6 bytes
            AS path: 3
                AS path segment: 3
                    Path segment type: AS_SEQUENCE (2)
                    Path segment length: 1 AS
                    Path segment value: 3
        MULTI_EXIT_DISC: 0 (7 bytes)
            Flags: 0x80 (Optional, Non-transitive, Complete)
                1... .... = Optional: Optional
                .0.. .... = Transitive: Non-transitive
                ..0. .... = Partial: Complete
                ...0 .... = Length: Regular length
            Type code: MULTI_EXIT_DISC (4)
```

```
              Length: 4 bytes
              Multiple exit discriminator: 0
          PREFIX-SID (13 bytes)
              Flags: 0xc0 (Optional, Transitive, Complete)
                  1... .... = Optional: Optional
                  .1.. .... = Transitive: Transitive
                  ..0. .... = Partial: Complete
                  ...0 .... = Length: Regular length
              Type code: PREFIX-SID (40)
              Length: 10 bytes
                  Label-index: 3 (1)
                      Type code: Label-index (1)
                      Length: 7 bytes
                      Flags: 0x00
                      Label index: 3
```

Node21 installs prefix 1.1.1.3/32 in the BGP table. The BGP table entry is displayed in Example 6-13. The different elements in the BGP update message are shown in this output: the AS-path attribute {3} in line 14, the BGP nexthop 99.3.21.3 on line 15, and the Origin and MED on line 17.

Node3 advertises label value “3” for this prefix, which is the implicit-null label (`Received Label 3` on line 16), and Node3 sets the Prefix-SID Label Index “3” (line 21), which is the value that was assigned to this prefix in the route-policy SID() configuration of Node3. Since Node21 has SR BGP enabled, it allocates a local label 16003 from the SRGB [16000-23999] for this prefix (16003 = 16000 + 3). This local label is displayed in line 6.

Example 6-13: BGP table entry of prefix 1.1.1.3/32 on Node21

```
1  RP/0/0/CPU0:xrvr-21#show bgp ipv4 labeled-unicast 1.1.1.3/32
2  BGP routing table entry for 1.1.1.3/32
3  Versions:
4    Process           bRIB/RIB  SendTblVer
5    Speaker                  3           3
6      Local Label: 16003
7  Last Modified: Aug 10 12:14:05.225 for 1d05h
```

```
 8| Paths: (1 available, best #1)
 9|   Advertised to update-groups (with more than one peer):
10|     0.2
11|   Path #1: Received by speaker 0
12|   Advertised to update-groups (with more than one peer):
13|     0.2
14|   3
15|     99.3.21.3 from 99.3.21.3 (1.1.1.3)
16|       Received Label 3
17|       Origin IGP, metric 0, localpref 100, valid, external,
best, group-best
18|       Received Path ID 0, Local Path ID 0, version 3
19|       Origin-AS validity: not-found
20|       Prefix SID Attribute Size: 10
21|       Label Index: 3
```

Node21 installs the forwarding entry for prefix 1.1.1.3/32 in its forwarding table. The RIB entry is shown in Example 6-14. The local label for prefix 1.1.1.3/32 is the Prefix-SID label 16003 (line 16). The label value 0x100004 (1048580) in the output (line 10) is a special internal value that represents the pop or implicit-null label. By default Cisco IOS XR automatically tags a BGP-learned route with the most recent AS number in its AS path (line 5); this is not related to SR BGP.

Example 6-14: RIB entry of prefix 1.1.1.3/32 on Node21

```
 1| RP/0/0/CPU0:xrvr-21#show route 1.1.1.3/32 detail
 2|
 3| Routing entry for 1.1.1.3/32
 4|   Known via "bgp 20", distance 20, metric 0, [ei]-bgp,
labeled unicast (3107), labeled SR
 5|   Tag 3, type external
 6|   Installed May 30 17:59:39.985 for 00:16:33
 7|   Routing Descriptor Blocks
 8|     99.3.21.3, from 99.3.21.3, BGP external
 9|       Route metric is 0
10|       Label: 0x100004 (1048580)
11|       Tunnel ID: None
12|       Binding Label: None
13|       Extended communities count: 0
```

```
        NHID:0x0(Ref:0)
    Route version is 0x2 (2)
    Local Label: 0x3e83 (16003)
    IP Precedence: Not Set
    QoS Group ID: Not Set
    Flow-tag: Not Set
    Fwd-class: Not Set
    Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type
RIB_SVD_TYPE_LOCAL
    Download Priority 4, Download Version 1802
    No advertising protos.
```

Node21 installs the label imposition entry in the cef table (see Example 6-15) and the label swap entry in the MPLS forwarding table (see Example 6-16). Node21 imposes no labels on incoming unlabeled packets destined for 1.1.1.3/32 since the destination Node3 is adjacent (last line in Example 6-15). The label is popped for labeled packets that arrive with top label 16003 (last line in Example 6-16).

Example 6-15: CEF entry of prefix 1.1.1.3/32 on Node21

```
RP/0/0/CPU0:xrvr-21#show cef 1.1.1.3/32
1.1.1.3/32, version 1802, internal 0x5000001 0x80 (ptr
0xa1492df4) [1], 0x0 (0xa145dc44), 0xa00 (0xa15832f8)
 Updated May 30 17:59:40.005
 Prefix Len 32, traffic index 0, precedence n/a, priority 4
   via 99.3.21.3/32, 3 dependencies, recursive, bgp-ext [flags
0x6020]
    path-idx 0 NHID 0x0 [0xa1492374 0xa15eb9f4]
    recursion-via-/32
    next hop 99.3.21.3/32 via 99.3.21.3/32
     local label 16003
     next hop 99.3.21.3/32 Gi0/0/0/0    labels imposed
{ImplNull ImplNull}
```

Example 6-16: MPLS forwarding table entry of prefix 1.1.1.3/32 on Node21

```
RP/0/0/CPU0:xrvr-21#show mpls forwarding prefix 1.1.1.3/32
Local  Outgoing    Prefix             Outgoing     Next
Hop          Bytes
```

```
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16003  Pop         SR Pfx (idx 3)     Gi0/0/0/0
99.3.21.3       5311
```

The BGP configuration of Node21 is shown in Example 6-17. The configuration of the different route-policies is the same as on Node3. Node21 has an AS number 21. Two neighbor-groups are configured: one for the BGP sessions to the Tier-1 nodes (`neighbor-group TIER1` on lines 22 to 25) and one for the EBGP sessions to the Tier-3 nodes (`neighbor-group TIER3` on lines 28 to 31). In this example, the two neighbor groups contain the same configuration. Node21 originates its loopback0 prefix 1.1.1.21/32 with a Prefix-SID index 21 to all its EBGP-LU neighbors (the `network` statement on line 19). The respective neighbor-group is applied to each neighbor. SRGB [16000-23999] is globally configured on lines 45 and 46.

Example 6-17: BGP configuration of Node21

```
 1  interface Loopback0
 2   ipv4 address 1.1.1.21 255.255.255.255
 3  !
 4  route-policy SID($SID)
 5    set label-index $SID
 6  end-policy
 7  !
 8  route-policy bgp_in
 9    pass
10  end-policy
11  !
12  route-policy bgp_out
13    pass
14  end-policy
15  !
16  router bgp 21
```

```
 bgp router-id 1.1.1.21
 address-family ipv4 unicast
  network 1.1.1.21/32 route-policy SID(21)
  allocate-label all
 !
 neighbor-group TIER1
  address-family ipv4 labeled-unicast
   route-policy bgp_in in
   route-policy bgp_out out
  !
 !
 neighbor-group TIER3
  address-family ipv4 labeled-unicast
   route-policy bgp_in in
   route-policy bgp_out out
  !
 !
 neighbor 99.3.21.3
  remote-as 3
  use neighbor-group TIER1
  description eBGP peer xrvr-3
 !
 neighbor 99.21.31.31
  remote-as 31
  use neighbor-group TIER3
  description eBGP peer xrvr-31
 !
!
segment-routing
 global-block 16000 23999
!
```

Node21 re-advertises prefix 1.1.1.3/32 to Node31. The captured BGP update message is displayed in Example 6-18. The MP_REACH_NRLI attribute is displayed in lines 8 to 25. Node21 has updated the BGP nexthop to its local interface address 99.21.31.21 (line 19) due to the default EBGP “next-hop-self” behavior. Node21 advertises the IPv4 Labeled Unicast prefix 1.1.1.3/32 with label value 16003 (lines 22 and 24); this is the local label that Node21 allocated for prefix 1.1.1.3/32. Node21 prepended its AS number 21 to the AS-path attribute (line 35). The Prefix-SID attribute is propagated unchanged (lines 48 to 60).

Example 6-18: BGP Update message sent by Node21

```
 1| Border Gateway Protocol - UPDATE Message
 2|     Marker: ffffffffffffffffffffffffffffffff
 3|     Length: 74
 4|     Type: UPDATE Message (2)
 5|     Unfeasible routes length: 0 bytes
 6|     Total path attribute length: 51 bytes
 7|     Path attributes
 8|         MP_REACH_NLRI (21 bytes)
 9|             Flags: 0x90 (Optional, Non-transitive, Complete, 
Extended Length)
10|                 1... .... = Optional: Optional
11|                 .0.. .... = Transitive: Non-transitive
12|                 ..0. .... = Partial: Complete
13|                 ...1 .... = Length: Extended length
14|             Type code: MP_REACH_NLRI (14)
15|             Length: 17 bytes
16|             Address family: IPv4 (1)
17|             Subsequent address family identifier: Labeled 
Unicast (4)
18|             Next hop network address (4 bytes)
19|                 Next hop: 99.21.31.21 (4)
20|             Subnetwork points of attachment: 0
21|             Network layer reachability information (8 bytes)
22|                 Label Stack=16003 (bottom) IPv4=1.1.1.3/32
23|                     MP Reach NLRI Prefix length: 56
24|                     MP Reach NLRI Label Stack: 16003 
(bottom)
25|                     MP Reach NLRI IPv4 prefix: 1.1.1.3 
(1.1.1.3)
26|         ORIGIN: IGP (4 bytes)
27|             Flags: 0x40 (Well-known, Transitive, Complete)
28|                 0... .... = Optional: Well-known
29|                 .1.. .... = Transitive: Transitive
30|                 ..0. .... = Partial: Complete
31|                 ...0 .... = Length: Regular length
32|             Type code: ORIGIN (1)
33|             Length: 1 byte
34|             Origin: IGP (0)
```

```
        AS_PATH: 21 3 (13 bytes)
            Flags: 0x40 (Well-known, Transitive, Complete)
                0... .... = Optional: Well-known
                .1.. .... = Transitive: Transitive
                ..0. .... = Partial: Complete
                ...0 .... = Length: Regular length
            Type code: AS_PATH (2)
            Length: 10 bytes
            AS path: 21 3
                AS path segment: 21 3
                    Path segment type: AS_SEQUENCE (2)
                    Path segment length: 2 ASs
                    Path segment value: 21 3
        PREFIX-SID (13 bytes)
            Flags: 0xc0 (Optional, Transitive, Complete)
                1... .... = Optional: Optional
                .1.. .... = Transitive: Transitive
                ..0. .... = Partial: Complete
                ...0 .... = Length: Regular length
            Type code: PREFIX-SID (40)
            Length: 10 bytes
                Label-index: 3 (1)
                    Type code: Label-index (1)
                    Length: 7 bytes
                    Flags: 0x00
                    Label index: 3
```

The BGP table entry for prefix 1.1.1.3/32 on Node31 is shown in Example 6-19. The label that Node31 received from Node21 is 16003 (line 16) and the SID index in the Prefix-SID attribute (lines 20 and 21) is 3. Since Node31 is SR BGP enabled, it allocates a local label 16003 from the SRGB for prefix 1.1.1.3/32 (line 6). Node31 uses the received SID index 3 to determine which local Prefix-SID label to allocate.

Example 6-19: BGP table entry of prefix 1.1.1.3/32 on Node31

```
1 RP/0/0/CPU0:xrvr-31#show bgp ipv4 labeled-unicast 1.1.1.3/32
2 BGP routing table entry for 1.1.1.3/32
3 Versions:
4   Process           bRIB/RIB  SendTblVer
```

```
  Speaker                    4           4
    Local Label: 16003
Last Modified: May 30 17:59:44.268 for 00:04:48
Paths: (1 available, best #1)
  Advertised to update-groups (with more than one peer):
    0.2
  Path #1: Received by speaker 0
  Advertised to update-groups (with more than one peer):
    0.2
  20 3
    99.21.31.21 from 99.21.31.21 (1.1.1.21)
      Received Label 16003
      Origin IGP, localpref 100, valid, external, best,
group-best
      Received Path ID 0, Local Path ID 0, version 4
      Origin-AS validity: not-found
      Prefix SID Attribute Size: 10
      Label Index: 3
```

Node31 installs the forwarding entries for prefix 1.1.1.3/32. The RIB entry is displayed in Example 6-20. The local label 16003 is shown in line 16 and the outgoing label 16003 is shown in line 10.

Example 6-20: RIB entry of prefix 1.1.1.3/32 on Node31

```
 1 RP/0/0/CPU0:xrvr-31#show route 1.1.1.3/32 detail
 2
 3 Routing entry for 1.1.1.3/32
 4   Known via "bgp 31", distance 20, metric 0, [ei]-bgp,
labeled unicast (3107), labeled SR
 5   Tag 21, type external
 6   Installed Aug 11 20:58:11.405 for 12:31:18
 7   Routing Descriptor Blocks
 8     99.21.31.21, from 99.21.31.21, BGP external
 9       Route metric is 0
10       Label: 0x3e83 (16003)
11       Tunnel ID: None
12       Binding Label: None
13       Extended communities count: 0
14       NHID:0x0(Ref:0)
15   Route version is 0x14 (20)
16   Local Label: 0x3e83 (16003)
```

```
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: Not Set
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type
RIB_SVD_TYPE_LOCAL
  Download Priority 4, Download Version 199
  No advertising protos.
```

The CEF entry for prefix 1.1.1.3/32 is shown in Example 6-21. The imposed label stack consists of two entries (shown in the last line of the output) because it is a recursive entry; prefix 1.1.1.3/32 resolves on nexthop 99.21.31.21. The first label value in the `labels imposed {ImpNull 16003}` list is the label to reach the nexthop, which is the directly connected Node21, hence the label `ImplNull`. The second label is the Prefix-SID label for prefix 1.1.1.3/32.

Example 6-21: CEF entry of prefix 1.1.1.3/32 on Node31

```
RP/0/0/CPU0:xrvr-31#show cef 1.1.1.3/32
1.1.1.3/32, version 213, internal 0x5000001 0x80 (ptr
0xa141aaf4) [1], 0x0 (0xa13e5974), 0xa08 (0xa1583230)
 Updated Aug 12 09:32:04.036
 Prefix Len 32, traffic index 0, precedence n/a, priority 4
   via 99.21.31.21/32, 5 dependencies, recursive, bgp-ext
[flags 0x6020]
    path-idx 0 NHID 0x0 [0xa15eb5f4 0x0]
    recursion-via-/32
    next hop 99.21.31.21/32 via 24001/0/21
     local label 16003
     next hop 99.21.31.21/32 Gi0/0/0/2    labels imposed
{ImplNull 16003}
```

The MPLS forwarding entry is shown in Example 6-22. Both local and outgoing labels for prefix 1.1.1.3/32 are 16003.

Example 6-22: MPLS forwarding table entry of prefix 1.1.1.3/32 on Node31

```
RP/0/0/CPU0:xrvr-31#show mpls forwarding prefix 1.1.1.3/32
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16003  16003       SR Pfx (idx 3)     Gi0/0/0/2
99.21.31.21     208
```

The prefix, Label-index and label value are further re-advertised by Node31 and Node11 to eventually arrive on Node1. The BGP configurations of Node11 and Node1 are equivalent to the ones above. Node1 then installs the forwarding entry for prefix 1.1.1.3/32 that is displayed in Example 6-23. Node1 is SR BGP enabled, hence it allocates a local label 16003 from the SRGB for prefix 1.1.1.3/32. This local label is installed in the MPLS forwarding entry that is displayed in Example 6-25.

Example 6-23: CEF entry of prefix 1.1.1.3/32 on Node1

```
RP/0/0/CPU0:xrvr-1#show cef 1.1.1.3/32
1.1.1.3/32, version 5107, internal 0x1000001 0x80 (ptr
0xa12753f4) [1], 0x0 (0xa123fd40), 0xa08 (0xa13ab0f0)
 Updated May 30 18:52:20.338
 Prefix Len 32, traffic index 0, precedence n/a, priority 4
   via 99.1.11.11/32, 5 dependencies, recursive, bgp-ext [flags
0x6020]
    path-idx 0 NHID 0x0 [0xa1413e74 0x0]
    recursion-via-/32
    next hop 99.1.11.11/32 via 24000/0/21
     local label 16003
     next hop 99.1.11.11/32 Gi0/0/0/0    labels imposed
{ImplNull 16003}
```

Example 6-24: MPLS forwarding table entry of prefix 1.1.1.3/32 on Node1

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding prefix 1.1.1.3/32
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
```

```
Label   Label        or ID
Interface                       Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16003  16003       SR Pfx (idx 3)     Gi0/0/0/0
99.1.11.11      0
```

Traceroute can be used to verify the path to 1.1.1.3 on Node1. The output of IP traceroute is shown in Example 6-25. The traced path is Node1→Node11→Node31→Node21→Node3. The same Prefix-SID label 16003 is used for all hops on the path, until penultimate hop Node21 pops the label. The MPLS traceroute output is displayed in Example 6-26. The outgoing label stack for each hop (`implicit-null/16003`) is the label stack as shown in the CEF entry in Example 6-23.

Example 6-25: IP traceroute from Node1 to Node3

```
RP/0/0/CPU0:xrvr-1#traceroute 1.1.1.3 source 1.1.1.1

Type escape sequence to abort.
Tracing the route to 1.1.1.3

 1  99.1.11.11 [MPLS: Label 16003 Exp 0] 289 msec  39 msec  39
msec
 2  99.11.31.31 [MPLS: Label 16003 Exp 0] 309 msec  49 msec  59
msec
 3  99.21.31.21 [MPLS: Label 16003 Exp 0] 199 msec  49 msec  69
msec
 4  99.3.21.3 299 msec  229 msec  219 msec
```

Example 6-26: MPLS traceroute from Node1 to Node3

```
RP/0/0/CPU0:xrvr-1#traceroute mpls ipv4 1.1.1.3/32 source
1.1.1.1 fec-type generic

Tracing MPLS Label Switched Path to 1.1.1.3/32, timeout is 2
seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output
interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC
mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx
label,
  'P' - no rx intf label prot, 'p' - premature termination of
LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

  0 99.1.11.1 MRU 1500 [Labels: implicit-null/16003 Exp: 0/0]
L 1 99.1.11.11 MRU 1500 [Labels: implicit-null/16003 Exp: 0/0]
20 ms
L 2 99.11.31.31 MRU 1500 [Labels: implicit-null/16003 Exp: 0/0]
210 ms
L 3 99.21.31.21 MRU 1500 [Labels: implicit-null/implicit-null
Exp: 0/0] 200 ms
! 4 99.3.21.3 220 ms
```

6.8.2 Configuring L3VPN Overlay Service

So far, this section discussed the SR MPLS transport across the Data Center Fabric. Next, an L3VPN overlay service is deployed between the two Tier-1 nodes Node1 and Node3. Servers or Virtual Machines that connect to these Tier-1 nodes' VRF interface then participate in the L3VPN service.

The vpnv4 prefix and label BGP exchange between the Tier-1 nodes is done via a Service Route Reflector Node41. In this example, the Service Route Reflector is connected to the Tier-3 Node31.

The L3VPN service configuration of Tier-1 Node3 is shown in Example 6-27. The vrf "SVC" is configured on lines 1 to 7, specifying the Route-target 1:1000 for both import and export. The address-family vpnv4 unicast is enabled under the BGP instance. A neighbor-group ROUTE_REFLECTORS is configured on lines 14 to 19, with the BGP neighbor configuration for the session to the Route Reflector Node41. The Route Reflector has an AS number 41. Node3 connects to it using a multi-hop EBGP session established between the loopbacks of both nodes. The transport between the loopback prefixes is SR MPLS. The vrf SVC is configured under BGP with an automatically selected Route Distinguisher (lines 27 to 30).

Example 6-27: BGP configuration for L3VPN service on Node3

```
 1| vrf SVC
 2|  address-family ipv4 unicast
 3|   import route-target
 4|    1:1000
 5|   !
 6|   export route-target
 7|    1:1000
 8|   !
 9|  !
10| !
11| router bgp 3
12|  address-family vpnv4 unicast
13|  !
14|  neighbor-group ROUTE_REFLECTORS
15|   ebgp-multihop 255
16|   update-source Loopback0
17|   address-family vpnv4 unicast
18|    route-policy bgp_in in
19|    route-policy bgp_out out
20|   !
21|  !
22|  neighbor 1.1.1.41
23|   remote-as 41
24|   use neighbor-group ROUTE_REFLECTORS
25|   description service RR Node41
26|  !
27|  vrf SVC
28|   rd auto
29|   address-family ipv4 unicast
```

```
   network 99.3.9.0/24
  !
 !
!
```

The BGP configuration for the SR MPLS underlay on Route Reflector Node41 is equivalent to the configurations of the other nodes in the Data Center Fabric. Example 6-28 shows the BGP configuration on RR Node41 for its function as L3VPN overlay service RR.

Example 6-28: BGP configuration for L3VPN service on Service RR Node41

```
router bgp 41
 address-family vpnv4 unicast
  retain route-target all
 !
 neighbor-group RR_CLIENTS
  ebgp-multihop 255
  update-source Loopback0
  address-family vpnv4 unicast
   route-policy bgp_in in
   route-policy bgp_out out
   next-hop-unchanged
  !
 !
 neighbor 1.1.1.1
  remote-as 1
  use neighbor-group RR_CLIENTS
 !
 neighbor 1.1.1.3
  remote-as 3
  use neighbor-group RR_CLIENTS
 !
!
```

By default, a node does not accept any VPN prefixes that do not match a local Route-target import policy. This default behavior is for scalability reasons to not waste memory for BGP entries that are not used anyway. (IBGP) Route Reflectors (RRs) have this filter disabled by default since their function is to reflect all VPN prefixes even if no vrf is configured on the RR. Node41 in this example is not a *real* RR in the strict sense since "Route Reflector" is an Internal BGP (IBGP) mechanism. Node41 on the other hand re-advertises or *reflects* the BGP updates because of native EBGP functionality. Since Node41 is not a *real* RR, the RT filter is not disabled by default. But the retention of VPN prefixes on Node41 can be achieved by configuring `retain route-target` under the vpnv4 address-family (see line 3 in Example 6-28). This configuration accepts the keyword `all` or a route-policy; `all` retains *all* prefixes while using a route-policy in the configuration allows filtering the prefixes that must be retained.

The RR's function is to re-advertise the VPN prefixes between its RR clients. An RR should not modify the BGP nexthop attribute when re-advertising a prefix. Therefore the default EBGP nexthop-self behavior is disabled by configuring `next-hop-unchanged` under the neighbor-group applied to the Route Reflector client sessions (line 11). This way the Route Reflector maintains the original BGP nexthop for the prefix when re-advertising it.

The BGP entry of the vrf SVC prefix 99.3.9.0/24 on Node1 is shown in Example 6-29. The BGP nexthop for the prefix is 1.1.1.3, which is the router-id of Node3 that originated the prefix (line 12). The label 90039 (line 13) is the Aggregate label that Node3 has allocated and advertised for vrf SVC. By default, a PE allocates and advertises a per-vrf Aggregate label for locally originated vrf prefixes. The PE does a *Pop-and-lookup* in the vrf forwarding table when a packet with such an Aggregate label arrives.

Example 6-29: BGP table entry of prefix 99.3.9.0/24 in vrf SVC on Node1

```
 1| RP/0/0/CPU0:xrvr-1#show bgp vrf SVC 99.3.9.0/24
 2| BGP routing table entry for 99.3.9.0/24, Route
Distinguisher: 1.1.1.1:1
 3| Versions:
 4|   Process           bRIB/RIB  SendTblVer
 5|   Speaker                 16          16
 6| Last Modified: May 30 18:52:19.880 for 10:45:06
 7| Paths: (1 available, best #1)
 8|   Not advertised to any peer
 9|   Path #1: Received by speaker 0
10|   Not advertised to any peer
11|   41 3
12|     1.1.1.3 from 1.1.1.41 (1.1.1.41)
13|       Received Label 90039
14|       Origin IGP, localpref 100, valid, external, best,
group-best, import-candidate, imported
15|       Received Path ID 0, Local Path ID 0, version 16
16|       Extended community: RT:1:1000
17|       Source AFI: VPNv4 Unicast, Source VRF: default, Source
Route Distinguisher: 1.1.1.3:1
```

The output in Example 6-30 shows the RIB entry for prefix 99.3.9.0/24 in vrf SVC on Node1. It shows that this vrf prefix resolves on its BGP nexthop 1.1.1.3 in the global (`default`) forwarding table (lines 8 and 9).

Example 6-30: RIB entry of prefix 99.3.9.0/24 in vrf SVC on Node1

```
 1| RP/0/0/CPU0:xrvr-1#show route vrf SVC 99.3.9.0/24
 2|
 3| Routing entry for 99.3.9.0/24
 4|   Known via "bgp 1", distance 20, metric 0
 5|   Tag 41, type external
 6|   Installed May 30 18:52:20.328 for 10:42:57
 7|   Routing Descriptor Blocks
 8|     1.1.1.3, from 1.1.1.41, BGP external
 9|       Nexthop in Vrf: "default", Table: "default", IPv4
Unicast, Table Id: 0xe0000000
10|       Route metric is 0
11|   No advertising protos.
```

The cef entry for this prefix in vrf SVC on Node1 (Example 6-33), shows the label stack that Node1 imposes on packets destined for that prefix. The label stack consists of three labels: `labels imposed {ImplNull 16003 90039}` (last line in output). The first two labels are the Segment Routing BGP-LU labels used to reach the BGP nexthop 1.1.1.3. This two-label stack is also shown in the output of `show cef 1.1.1.3/32` on Node1 in Example 6-23. The first of these two labels is the label to reach the nexthop 99.1.11.11/32. Since this nexthop is directly connected, no label is required, hence the `ImplNull` label. The second label is the Prefix-SID label 16003 for prefix 1.1.1.3/32. The last label of the three-label stack, the label at the bottom of the stack, is the VPN label 90039 used for this prefix.

Example 6-31: CEF entry of prefix 99.3.9.0/24 in vrf SVC on Node1

```
RP/0/0/CPU0:xrvr-1#show cef vrf SVC 99.3.9.0/24
99.3.9.0/24, version 5, internal 0x5000001 0x0 (ptr 0xa1273974)
[1], 0x0 (0x0), 0x208 (0xa13ab2d0)
 Updated May 30 18:52:20.338
 Prefix Len 24, traffic index 0, precedence n/a, priority 3
   via 1.1.1.3/32, 3 dependencies, recursive, bgp-ext [flags
0x6020]
    path-idx 0 NHID 0x0 [0xa1413df4 0x0]
    recursion-via-/32
    next hop VRF - 'default', table - 0xe0000000
    next hop 1.1.1.3/32 via 16003/0/21
     next hop 99.1.11.11/32 Gi0/0/0/0    labels imposed
{ImplNull 16003 90039}
```

An IP traceroute in vrf SVC on Node1 to destination 99.3.9.3 is shown in Example 6-32. The packets travel via Node11, Node31, and Node21 to arrive on Node3. The labels on the packets are the Prefix-SID label 16003 and the VPN label 90039. A small detail: the last hop in the traceroute does not show the VPN label since the destination prefix of the traceroute is local to Node3. If the destination address would be a CE connected on the vrf interface then the VPN label would be shown as the only label for the fourth hop in the traceroute.

Example 6-32: IP traceroute in vrf SVC from Node1 to destination in vrf SVC

```
RP/0/0/CPU0:xrvr-1#traceroute vrf SVC 99.3.9.3

Type escape sequence to abort.
Tracing the route to 99.3.9.3

 1  99.1.11.11 [MPLS: Labels 16003/90039 Exp 0] 879 msec  49
msec  79 msec
 2  99.11.31.31 [MPLS: Labels 16003/90039 Exp 0] 39 msec  49
msec  39 msec
 3  99.21.31.21 [MPLS: Labels 16003/90039 Exp 0] 49 msec  39
msec  39 msec
 4  99.3.21.3 59 msec  39 msec  39 msec
```

6.8.3 Configuring ECMP with E-BGP Multipath

When adding the other of the nodes of the topology in Figure 6-12 (Node2, Node12, Node32, Node22, and Node4), multiple Equal Cost paths become available in the Data Center Fabric, also between Node1 and Node3. Of course, all available ECMP must be used in the Data Center in order to use all available network capacity. By default, BGP selects one path to each destination as best path, and only that path is installed in the forwarding table. To install multiple paths per destination, BGP multi-path must be enabled. EBGP multi-path is enabled by configuring `maximum-paths ebgp <n>` under the address-family, where `<n>` specifies the maximum number of paths that will be installed for each destination. This configuration is shown in Example 6-31.

Example 6-33: EBGP multipath configuration

```
router bgp 1
```

```
  address-family ipv4 unicast
   maximum-paths ebgp 16   !! maximum 16 Equal Cost paths per
prefix
  !
 !
```

An additional configuration may be required, depending on the AS number assignment in the Data Center Fabric. Node1 in Figure 6-12 receives two EBGP-LU paths for Node3's loopback prefix 1.1.1.3/32 (see the output in Example 6-34): one via Node11 (lines 11 to 21) and one via Node12 (lines 22 to 31). Due to the AS number assignment in the example (each node has its own unique AS number, both paths do not have the same AS-path: {11 31 21 3} for the first path (line 14) and {12 31 21 3} for the second path (line 24). The path via Node11 is selected as the best path since all attributes are equal for both paths (only the length of AS-path counts in the best path evaluation) and the path via Node11 has the lowest neighbor address. The second path is not admitted in the multi-path since EBGP multi-path requires the candidate paths to have a cost equal to the best path; i.e. have identical BGP attributes: same Weight, same Local-Pref, same AS-Path (AS-path length and content), same Origin, same MED, and a different nexthop.

Example 6-34: BGP table entry of prefix 1.1.1.3/32 on Node1 with default EBGP multipath selection

```
 1  RP/0/0/CPU0:xrvr-1#show bgp ipv4 labeled-unicast 1.1.1.3/32
 2  BGP routing table entry for 1.1.1.3/32
 3  Versions:
 4    Process           bRIB/RIB  SendTblVer
 5    Speaker                 48          48
 6      Local Label: 16003
 7  Last Modified: Aug 12 11:52:24.342 for 00:01:55
 8  Paths: (2 available, best #1)
 9    Advertised to update-groups (with more than one peer):
10      0.2
11  11  Path #1: Received by speaker 0
12    Advertised to update-groups (with more than one peer):
```

```
13|     0.2
14|   11 31 21 3
15|     99.1.11.11 from 99.1.11.11 (1.1.1.11)
16|       Received Label 16003
17|       Origin IGP, localpref 100, valid, external, best,
group-best
18|       Received Path ID 0, Local Path ID 0, version 48
19|       Origin-AS validity: not-found
20|       Prefix SID Attribute Size: 10
21| 21        Label Index: 3
22| 22    Path #2: Received by speaker 0
23|   Not advertised to any peer
24|   12 31 21 3
25|     99.1.12.12 from 99.1.12.12 (1.1.1.12)
26|       Received Label 16003
27|       Origin IGP, localpref 100, valid, external, group-best
28|       Received Path ID 0, Local Path ID 0, version 0
29|       Origin-AS validity: not-found
30|       Prefix SID Attribute Size: 10
31| 31        Label Index: 3
```

To admit the second path in the multi-path, the multipath selection rule can be relaxed to also permit paths with the same AS-path length as the best path, but different AS-path content. Therefore, `bgp bestpath as-path multipath-relax` must be configured. This is illustrated in Example 6-35. Now the second path is accepted in the multi-path, as indicated in lines 23 and 33.

Example 6-35: BGP table entry of prefix 1.1.1.3/32 on Node1 with relaxed EBGP multipath selection

```
 1| RP/0/0/CPU0:xrvr-1#configure
 2| RP/0/0/CPU0:xrvr-1(config)#router bgp 1
 3| RP/0/0/CPU0:xrvr-1(config-bgp)# bgp bestpath as-path
multipath-relax
 4| RP/0/0/CPU0:xrvr-1(config-bgp)#commit
 5| RP/0/0/CPU0:xrvr-1(config-bgp)#end
 6| RP/0/0/CPU0:xrvr-1#
 7| RP/0/0/CPU0:xrvr-1#show bgp ipv4 labeled-unicast 1.1.1.3/32
 8| BGP routing table entry for 1.1.1.3/32
 9| Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 54          54
    Local Label: 16003
Last Modified: Aug 12 12:32:22.342 for 00:00:12
Paths: (2 available, best #1)
  Advertised to update-groups (with more than one peer):
    0.2
  Path #1: Received by speaker 0
  Advertised to update-groups (with more than one peer):
    0.2
  11 31 21 3
    99.1.11.11 from 99.1.11.11 (1.1.1.11)
      Received Label 16003
      Origin IGP, localpref 100, valid, external, best,
group-best, multipath
      Received Path ID 0, Local Path ID 0, version 54
      Origin-AS validity: not-found
      Prefix SID Attribute Size: 10
      Label Index: 3
  Path #2: Received by speaker 0
  Not advertised to any peer
  12 31 21 3
    99.1.12.12 from 99.1.12.12 (1.1.1.12)
      Received Label 16003
      Origin IGP, localpref 100, valid, external, multipath
      Received Path ID 0, Local Path ID 0, version 0
      Origin-AS validity: not-found
      Prefix SID Attribute Size: 10
      Label Index: 3
```

The equal cost paths are also added to the forwarding table, as illustrated in Example 6-36 and Example 6-37.

Example 6-36: CEF entry of prefix 1.1.1.3/32 on Node1

```
RP/0/0/CPU0:xrvr-1#show cef 1.1.1.3/32
1.1.1.3/32, version 305, internal 0x1000001 0x80 (ptr
0xa141a6f4) [1], 0x0 (0xa13e5758), 0xa08 (0xa16f0050)
 Updated Aug 12 12:32:22.675
 Prefix Len 32, traffic index 0, precedence n/a, priority 4
   via 99.1.11.11/32, 5 dependencies, recursive, bgp-ext, bgp-
multipath [flags 0x60a0]
```

```
    path-idx 0 NHID 0x0 [0xa15eb4f4 0x0]
    recursion-via-/32
    next hop 99.1.11.11/32 via 24000/0/21
     local label 16003
     next hop 99.1.11.11/32 Gi0/0/0/0    labels imposed
{ImplNull 16003}
   via 99.1.12.12/32, 5 dependencies, recursive, bgp-ext, bgp-
multipath [flags 0x60a0]
    path-idx 1 NHID 0x0 [0xa15ebbf4 0x0]
    recursion-via-/32
    next hop 99.1.12.12/32 via 24007/0/21
     local label 16003
     next hop 99.1.12.12/32 Gi0/0/0/1    labels imposed
{ImplNull 16003}
```

Example 6-37: MPLS forwarding table entry of prefix 1.1.1.3/32 on Node1

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding prefix 1.1.1.3/32
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16003  16003       SR Pfx (idx 3)     Gi0/0/0/0
99.1.11.11      0
       16003       SR Pfx (idx 3)     Gi0/0/0/1
99.1.12.12      0
```

MPLS multi-path traceroute can be used to explore the number of possible paths from Node1 to Node3 through the Data Center fabric. Chapter 11, “Verifying connectivity in SR MPLS network” describes the MPLS multi-path traceroute functionality in more detail. The output of the `traceroute mpls` command is displayed in Example 6-38. Eight different paths are found to go from Node1 to Node3 in this topology.

Example 6-38: MPLS multipath traceroute from Node1 to Node3

```
RP/0/0/CPU0:xrvr-1#traceroute mpls multipath ipv4 1.1.1.3/32
source 1.1.1.1 fec-type generic
```

```
Starting LSP Path Discovery for 1.1.1.3/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output
interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC
mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx
label,
  'P' - no rx intf label prot, 'p' - premature termination of
LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

LLL!
Path 0 found,
 output interface GigabitEthernet0/0/0/0 nexthop 99.1.11.11
 source 1.1.1.1 destination 127.0.0.1
L!
Path 1 found,
 output interface GigabitEthernet0/0/0/0 nexthop 99.1.11.11
 source 1.1.1.1 destination 127.0.0.0
LL!
Path 2 found,
 output interface GigabitEthernet0/0/0/0 nexthop 99.1.11.11
 source 1.1.1.1 destination 127.0.0.6
L!
Path 3 found,
 output interface GigabitEthernet0/0/0/0 nexthop 99.1.11.11
 source 1.1.1.1 destination 127.0.0.3
LLL!
Path 4 found,
 output interface GigabitEthernet0/0/0/1 nexthop 99.1.12.12
 source 1.1.1.1 destination 127.0.0.1
L!
Path 5 found,
 output interface GigabitEthernet0/0/0/1 nexthop 99.1.12.12
 source 1.1.1.1 destination 127.0.0.0
LL!
Path 6 found,
 output interface GigabitEthernet0/0/0/1 nexthop 99.1.12.12
```

```
  source 1.1.1.1 destination 127.0.0.4
L!
Path 7 found,
  output interface GigabitEthernet0/0/0/1 nexthop 99.1.12.12
  source 1.1.1.1 destination 127.0.0.3

Paths (found/broken/unexplored) (8/0/0)
  Echo Request (sent/fail) (22/0)
  Echo Reply (received/timeout) (22/0)
  Total Time Elapsed 559 ms
```

A server/VM with IP address 99.3.9.9 is multi-homed to Node3 and Node4. Both Node3 and Node4 advertise their reachability to the prefix 99.3.9.0/24 in vrf SVC. Both nodes use a different Route Distinguisher (RD) for this prefix, the automatically assigned RDs 1.1.1.3:1 and 1.1.1.4:0 respectively. Because the RDs are different, the Route Reflector Node41 sees both routes as different routes and it reflects both routes to its RR clients. Node1 receives the two routes and imports them both in the vrf SVC BGP table. The BGP entries for vrf SVC prefix 99.3.9.0/24 are shown in Example 6-39. Only one of the paths is selected as best path and installed in the forwarding table; this is the second path, via Node4.

Example 6-39: BGP table entry of prefix 99.3.9.0/24 in vrf SVC on Node1 without multipath

```
RP/0/0/CPU0:xrvr-1#show bgp vrf SVC 99.3.9.0/24
BGP routing table entry for 99.3.9.0/24, Route Distinguisher:
1.1.1.1:1
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 27          27
Last Modified: Aug 12 10:30:12.342 for 02:26:45
Paths: (2 available, best #2)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  41 3
    1.1.1.3 from 1.1.1.41 (1.1.1.41)
      Received Label 90039
      Origin IGP, localpref 100, valid, external, import-
candidate, imported
      Received Path ID 0, Local Path ID 0, version 0
      Extended community: RT:1:1000
      Source AFI: VPNv4 Unicast, Source VRF: default, Source
Route Distinguisher: 1.1.1.3:1
  Path #2: Received by speaker 0
  Not advertised to any peer
  41 4
    1.1.1.4 from 1.1.1.41 (1.1.1.41)
      Received Label 90049
      Origin IGP, localpref 100, valid, external, best, group-
best, import-candidate, imported
      Received Path ID 0, Local Path ID 0, version 27
      Extended community: RT:1:1000
      Source AFI: VPNv4 Unicast, Source VRF: default, Source
Route Distinguisher: 1.1.1.4:0
```

To enable load-sharing between the two paths, BGP multi-path must be enabled under the vrf. See the configuration in Example 6-40. `maximum-paths ebgp 16` is configured under the `vrf SVC` `address-family ipv4 unicast`. Notice in the output of Example 6-39 that the two paths have a different AS-Path: {41 3} for the first path and {41 4} for the second path. Due to this, the non-best path is not eligible as an additional path for multipath. Therefore the `multipath-relax` configuration must be applied under the vrf (line 3 of Example 6-40) to permit the non-best path as an additional path in the BGP multi-path.

Example 6-40: BGP multipath configuration on Tier-1 nodes

```
1 vrf SVC
2   rd auto
3   bgp bestpath as-path multipath-relax
4   address-family ipv4 unicast
5    maximum-paths ebgp 16
6    network 99.3.9.0/24
7   !
8  !
```

```
!
```

The BGP multi-path load-balancing in the VPN service overlay adds another level of load-sharing to the network: load-sharing over the two Tier-1 nodes in the service overlay network and load-sharing over the Data Center fabric towards each Tier-1 node in the Data Center Fabric. Example 6-41 shows the two paths towards multi-homed prefix 99.3.9.0/24 in vrf SVC: the first path in lines 8 to 10 and the second path in lines 11 to 13.

Example 6-41: BGP table entry of prefix 99.3.9.0/24 in vrf SVC on Node1 with multipath

```
 1| RP/0/0/CPU0:xrvr-1#show route vrf SVC 99.3.9.0/24
 2|
 3| Routing entry for 99.3.9.0/24
 4|   Known via "bgp 1", distance 20, metric 0
 5|   Tag 40, type external
 6|   Installed Aug 12 10:30:12.648 for 02:27:11
 7|   Routing Descriptor Blocks
 8|     1.1.1.3, from 1.1.1.41, BGP external, BGP multi path
 9|       Nexthop in Vrf: "default", Table: "default", IPv4
Unicast, Table Id: 0xe0000000
10|       Route metric is 0
11|     1.1.1.4, from 1.1.1.41, BGP external, BGP multi path
12|       Nexthop in Vrf: "default", Table: "default", IPv4
Unicast, Table Id: 0xe0000000
13|       Route metric is 0
14|   No advertising protos.
```

6.8.4 Configuring I-BGP Prefix Segments

Internal BGP-LU (IBGP-LU) can be used as well in the Data Center. While the core of the BGP protocol is the same for EBGP and IBGP, there are differences in protocol behavior between the two and the desired BGP behavior for this application is closest to the EBGP behavior. Therefore, an EBGP-like behavior is applied to IBGP. This means:

- Use Route Reflector functionality to re-advertise routes between peers
- Use Cluster-list to prevent loops and assign Cluster-ids similar to AS numbers
- Use next-hop-self for re-advertised routes

REMINDER: IBGP Route Reflector functionality

To prevent loops inside an Autonomous System (AS), there is a rule that specifies that all IBGP nodes in the network must not re-advertise any route received from an IBGP neighbor to any other IBGP neighbor. Because of this rule, all IBGP nodes in the AS should be connected in a logical full mesh to allow route propagation to all IBGP nodes in the AS. However, a full mesh is not a scalable solution. One possible way to eliminate the need for a full mesh is to use Route Reflectors.

A Route Reflector (RR) is a BGP node that is allowed to advertise updates received from an IBGP peer to another IBGP peer. RRs are specified in IETF RFC 4456.

But there are certain rules under which an RR can reflect a route. For this purpose, the IBGP peers of an RR are divided into two groups: client peers and non-client peers. The rules:

- A route received from a client peer is reflected to its client peers and non-client peers.
- A route received from a non-client peer is reflected to client peers only.

When an RR reflects a route that does not already have an Originator-id, then the RR adds that attribute in the reflected update and sets it to the BGP router-id of the originating node. The RR also prepends its own Cluster-id to the Cluster-list attribute of the reflected route. A node ignores a received update with the Originator-id equal to its own Router-id. An RR ignores a received update that contains its local Cluster-id in the Cluster-list attribute. This is the loop prevention mechanism used when using RRs.

By default, a RR only reflects the BGP best path for a prefix.

When an RR reflects a route, it should not modify the next-hop attribute, among others. Sometimes next-hop-self behavior is required; therefore modification of the next-hop attribute is required. To *really* make sure attribute modifications are not done by accident, `ibgp policy out enforce-modifications` must be configured to allow modifying attributes of reflected routes.

RRs can at the same time be clients of other RRs, thus realizing a hierarchical RR setup.

A Route Reflector hierarchy can naturally be applied to a Clos topology. In a three-tier Clos topology, the Tier-3 nodes are the top-level RRs. The Tier-2 nodes are clients of the Tier-3 RRs and are RRs themselves towards the Tier-1 nodes. In other words, the lower tier nodes are RR clients of the higher tier nodes. The higher tier nodes are regular (non-RR client) peers of the lower tier nodes.

The route-reflection is illustrated in the network topology of Figure 6-21. For simplicity of the illustration, it is assumed the route becomes the BGP best path for the prefix on each node. The BGP update messages are shown in the unfolded partial topology in Figure 6-22. Node1 sends a BGP update message for its loopback prefix 1.1.1.1/32 to Node11. Since Node1 is a RR client of Node11, Node11 reflects the route to its RR clients Node1 and Node2 and to its non-RR client peers Node31 and Node32. Node11 sets the Originator-id attribute in the update message to Node1's router-id 1.1.1.1 and prepends its own cluster-id 1.1.1.11 to the Cluster-list attribute. Since Node1 sees its own BGP router-id in the Originator-id attribute, it ignores the update. Both Tier-3 nodes Node31 and Node32 act similarly on the received update message, but only the route reflection of Node31 is illustrated. Node31 reflects the route received from its RR client Node11 to its RR clients Node11, Node12, Node21, and Node22. Node11 sees it local cluster-id 11 in the Cluster-list, therefore it ignores this update message. The other three Tier-2 nodes act alike, but only the route reflection of Node21 is illustrated. Node21 receives the route from its non-RR client peer Node31 and only reflects the route to its RR client peers Node3 and Node4.

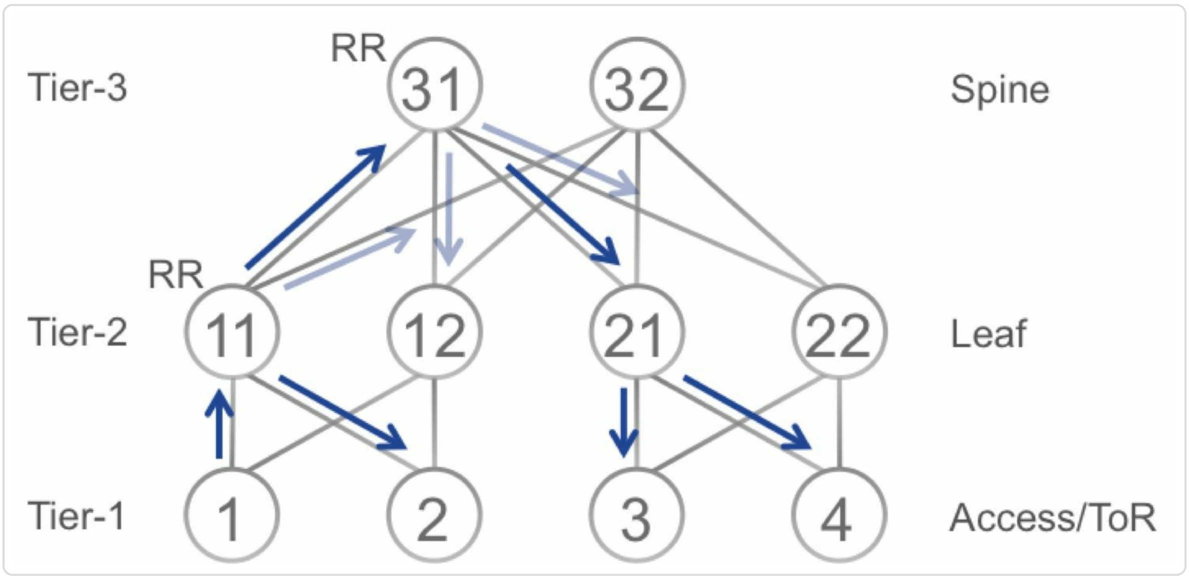


Figure 6-21: Route reflection in Clos topology using IBGP

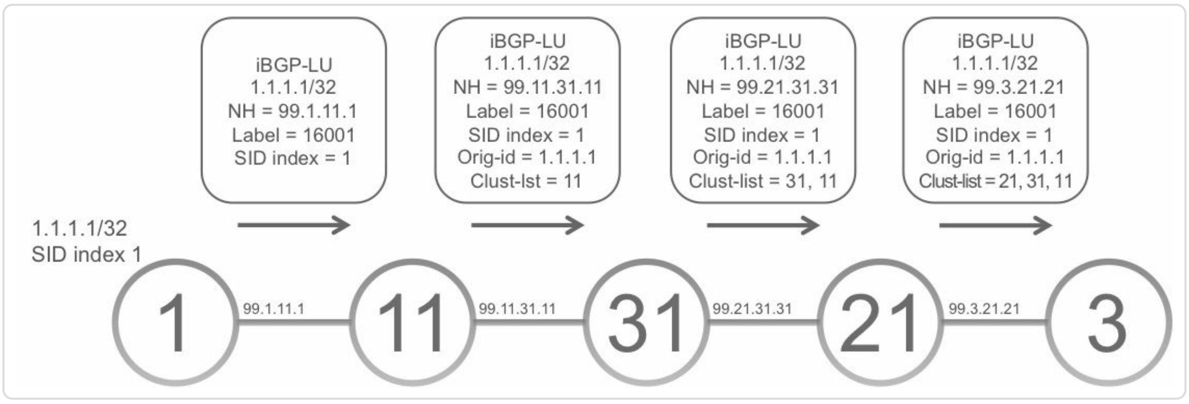


Figure 6-22: Unfolded representation of one path in the network topology

Same as for EBGP, BGP multi-path can be enabled for IBGP to load-balance over multiple equal cost paths. However, a limitation exists in the IBGP multipath selection. Remind the BGP selection process; BGP selects the best path, following the best path selection rules:

1. Highest weight
2. Highest Local Preference
3. Shortest AS Path Length
4. Lowest Origin Type

5. Lowest MED (Multi-Exit Discriminator)
6. Prefer External to Internal
7. Closest Egress (Lowest IGP Distance)
8. Lowest Router-ID (Tie breaker)
9. Shortest Cluster-list length (Tie breaker)
10. Lowest neighbor address (Tie breaker)

IBGP multipath selection stops at rule 7. BGP bestpath selection continues until rule 10. This means that all paths that are equal for the first 6 rules are eligible for multipath. Notice that rule 9 that prefers the shortest Cluster-list length is not considered for IBGP multipath. The consequence is that a node may consider less desirable paths for multipath. Using the example topology Figure 6-21, the BGP update messages that arrive on Node12 are illustrated in Figure 6-23, together with the BGP table entry for prefix 1.1.1.1/32 for some of the nodes. Only part of the topology is shown in Figure 6-23.

Node1 advertises prefix 1.1.1.1/32 to Node11 and Node12. Node11 reflects this prefix to Node31 and Node32. Node12 does the same, but this is not illustrated. Node31 and Node32 both select the path received from Node11 as their best path. The two paths are equal up to best path selection rule 9 (see above), therefore the best path is selected based on the lowest neighbor address, which is the one used by Node11. Node31 and Node32 both advertise their best path to Node12, hence Node12 has three usable paths towards 1.1.1.1/32. Out of these it selects the bestpath based on the shortest cluster-list (rule 9): the direct path via Node1. However, the other paths are eligible for multipath since they have the same attributes up to rule 7. This results in Node2 load-balancing traffic to 1.1.1.1/32 over three paths, all three BGP paths for 1.1.1.1/32. This is highly undesirable since it leads to non-optimal paths and packet loops.

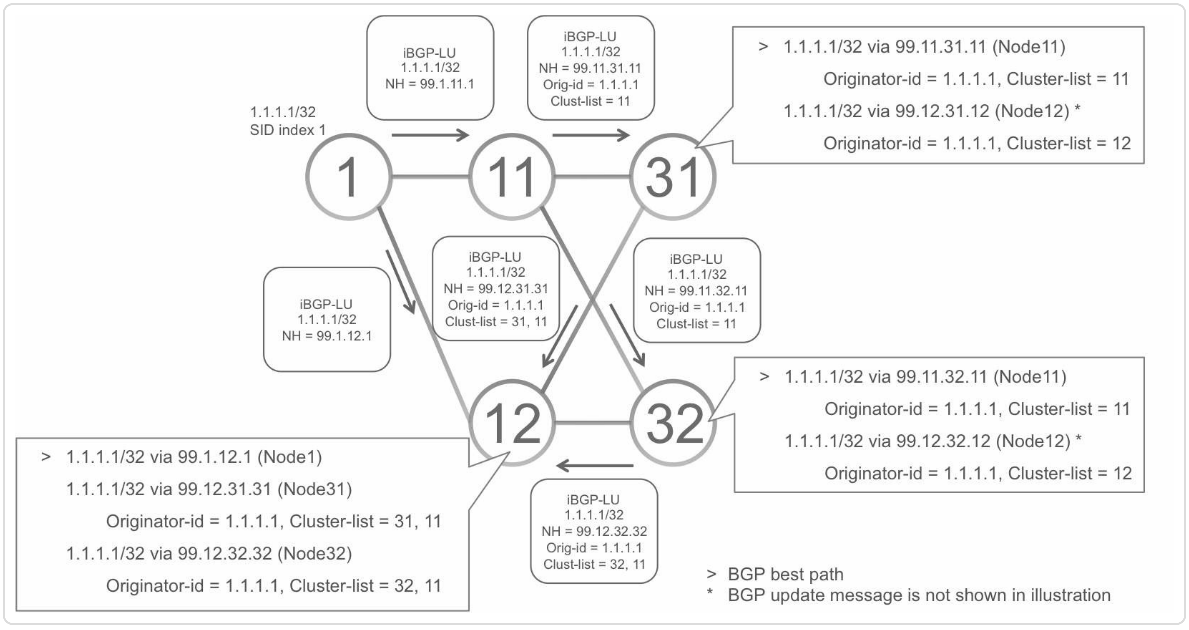

1.1.1.1/32
SID index 1
iBGP-LU 1.1.1.1/32 NH = 99.1.11.1
iBGP-LU 1.1.1.1/32 NH = 99.11.31.11 Orig-id = 1.1.1.1 Clust-list = 11
> 1.1.1.1/32 via 99.11.31.11 (Node11)
Originator-id = 1.1.1.1, Cluster-list = 11
1.1.1.1/32 via 99.12.31.12 (Node12) *
Originator-id = 1.1.1.1, Cluster-list = 12
1
11
31
iBGP-LU 1.1.1.1/32 NH = 99.1.12.1
iBGP-LU 1.1.1.1/32 NH = 99.12.31.31 Orig-id = 1.1.1.1 Clust-list = 31, 11
iBGP-LU 1.1.1.1/32 NH = 99.11.32.11 Orig-id = 1.1.1.1 Clust-list = 11
> 1.1.1.1/32 via 99.11.32.11 (Node11)
Originator-id = 1.1.1.1, Cluster-list = 11
1.1.1.1/32 via 99.12.32.12 (Node12) *
Originator-id = 1.1.1.1, Cluster-list = 12
12
32
> 1.1.1.1/32 via 99.1.12.1 (Node1)
1.1.1.1/32 via 99.12.31.31 (Node31)
Originator-id = 1.1.1.1, Cluster-list = 31, 11
1.1.1.1/32 via 99.12.32.32 (Node32)
Originator-id = 1.1.1.1, Cluster-list = 32, 11
iBGP-LU 1.1.1.1/32 NH = 99.12.32.32 Orig-id = 1.1.1.1 Clust-list = 32, 11
> BGP best path
* BGP update message is not shown in illustration

Figure 6-23: IBGP multi-path problem

Multiple solutions are possible for this problem. A possible solution could be to use the same cluster-id on the Tier-2 nodes in the same cluster. The last “cluster” in the previous sentence is used in the meaning of a set of directly connected Tier-1 and Tier-2 devices along with their attached servers. If Node11 and Node12 use the same cluster-id 10, then they ignore the routes that Node31 and Node32 reflect back to them. Node12 then only has a single route to 1.1.1.1/32: via Node1.

Another possible solution is to increment a metric every time a route is reflected. This metric could be any attribute that influences the bestpath selection. In this example, Accumulated IGP (AIGP) is used.

IETF RFC 7311 specifies AIGP. Using the AIGP attribute allows BGP to use IGP metric for bestpath selection, which eases BGP best path selection in a multi-AS network under a common administration. When using AIGP, the path with the lowest AIGP metric is preferred before considering the AS-path length of that path, so between the Local Preference and AS-path length steps in the BGP best path selection rules. Lacking IGP in a BGP-only network, the AIGP metric can be statically incremented instead.

BGP best path selection prefers the lowest AIGP metric path and only paths with the same AIGP metric as the BGP bestpath are eligible for multipath selection.

When incrementing the AIGP metric when advertising a route, a path is selected similar to shortest path selection in IGP.

The complete IBGP-LU configuration is analyzed based on the topology in Figure 6-24.

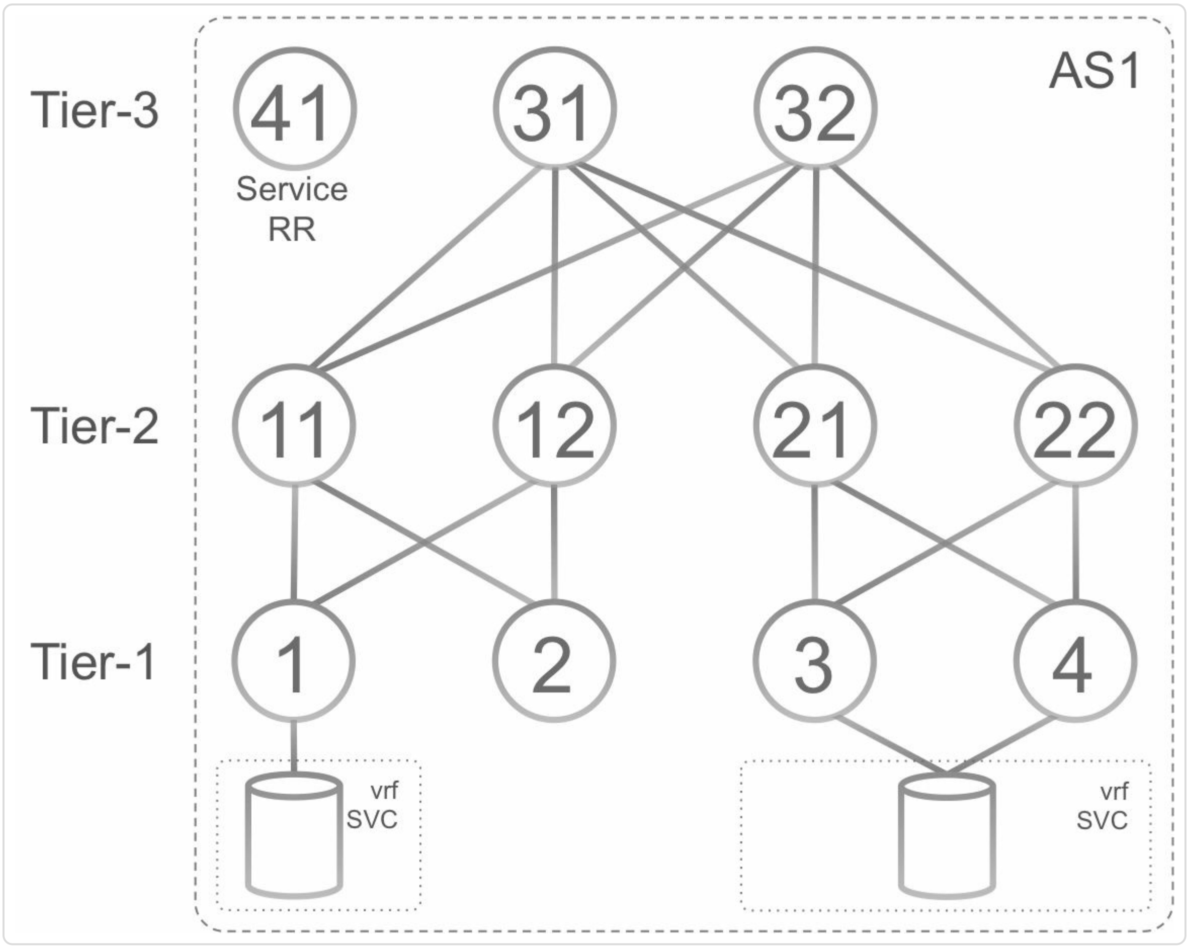


Figure 6-24: SR IBGP example network topology

The BGP configuration of the Tier-1 Node3 is shown in Example 6-42. The output starts with the route-policies that are used. `SID($SID)` to set the SID index to `$SID` and `ADDMETRIC($METRIC)` to increment the AIGP metric with value `$METRIC`. The RPL statement `set aigp-metric + $METRIC` adds `$METRIC` to the current `aigp-metric` value.

Similar to EBGP, the IBGP sessions are established between interface addresses, and a static route to the neighbor's interface address is required to resolve the BGP-LU routes. Contrary to EBGP-LU, MPLS is not enabled by default for IBGP-LU session between connected interfaces. MPLS is manually enabled on the peering interfaces by adding them under the `mpls activate` configuration (lines 12 to 14). Contrary to EBGP, IBGP does not require applying ingress and egress route-policies, only an egress route-policy is applied to increment the AIGP metric (see line 23 where AIGP metric is incremented by 10). IBGP multipath is enabled by `maximum-paths ibgp 16` on line 17.

Example 6-42: IBGP configuration of Node3

```
 1| route-policy SID($SID)
 2|   set label-index $SID
 3| end-policy
 4| !
 5| route-policy ADDMETRIC($METRIC)
 6|   set aigp-metric + $METRIC
 7|   pass
 8| end-policy
 9| !
10| router bgp 1
11|  bgp router-id 1.1.1.3
12|  mpls activate
13|   interface GigabitEthernet0/0/0/0
14|   interface GigabitEthernet0/0/0/1
15|  !
16|  address-family ipv4 unicast
17|   maximum-paths ibgp 16
18|   network 1.1.1.3/32 route-policy SID(3)
19|   allocate-label all
20|  !
21|  neighbor-group TIER2
22|   address-family ipv4 labeled-unicast
23|    route-policy ADDMETRIC(10) out
24|   !
25|  !
26|  neighbor 99.3.21.21
27|   remote-as 1
28|   use neighbor-group TIER2
29|   description iBGP peer xrvr-21
30|  !
31|  neighbor 99.3.22.22
32|   remote-as 1
33|   use neighbor-group TIER2
```

```
34   description iBGP peer xrvr-22
35  !
36 !
37 segment-routing
38  global-block 16000 23999
39 !
```

The BGP configuration of Leaf Node21 is shown in Example 6-43. Node21 has two neighbor-groups configured, one for the IBGP sessions to the Tier-1 nodes (`TIER1` on line 15) and one for the IBGP sessions to the Tier-3 nodes (`TIER3` on line 22). The Tier-1 neighbors are configured as RR clients (see line 17), and next-hop-self is configured for these sessions (see line 19). Router Reflectors should not modify BGP attributes such as the BGP Nexthop attribute. A safeguard command needs to be enabled to allow next-hop-self to be applied: `ibgp policy out enforce-modifications` on line 9. The Tier-3 neighbors are not configured as RR clients, but next-hop-self is enabled. For all neighbors, the AIGP metric of each route is incremented by 10 when advertising.

Example 6-43: IBGP configuration of Node21

```
 1 router bgp 1
 2  bgp router-id 1.1.1.21
 3  mpls activate
 4   interface GigabitEthernet0/0/0/0
 5   interface GigabitEthernet0/0/0/1
 6   interface GigabitEthernet0/0/0/2
 7   interface GigabitEthernet0/0/0/3
 8  !
 9  ibgp policy out enforce-modifications
10  address-family ipv4 unicast
11   maximum-paths ibgp 16
12   network 1.1.1.21/32 route-policy SID(21)
13   allocate-label all
14  !
15  neighbor-group TIER1
16   address-family ipv4 labeled-unicast
```

```
   route-reflector-client
   route-policy ADDMETRIC(10) out
   next-hop-self
  !
 !
 neighbor-group TIER3
  address-family ipv4 labeled-unicast
   route-policy ADDMETRIC(10) out
   next-hop-self
  !
 !
 neighbor 99.3.21.3
  remote-as 1
  use neighbor-group TIER1
  description iBGP peer xrvr-3
 !
 neighbor 99.4.21.4
  remote-as 1
  use neighbor-group TIER1
  description iBGP peer xrvr-4
 !
 neighbor 99.21.31.31
  remote-as 1
  use neighbor-group TIER3
  description iBGP peer xrvr-31
 !
 neighbor 99.21.32.32
  remote-as 1
  use neighbor-group TIER3
  description iBGP peer xrvr-32
 !
!
```

The BGP configuration of Node31 is shown in Example 6-44. Node31 has one neighbor-groups configured for the Tier-2 neighbors (`TIER2` on line 15). These neighbors are configured as Route Reflector clients and next-hop-self is enabled.

Example 6-44: IBGP configuration of Node31

```
1 router bgp 1
```

```
 bgp router-id 1.1.1.31
 mpls activate
  interface GigabitEthernet0/0/0/0
  interface GigabitEthernet0/0/0/1
  interface GigabitEthernet0/0/0/2
  interface GigabitEthernet0/0/0/3
 !
 ibgp policy out enforce-modifications
 address-family ipv4 unicast
  maximum-paths ibgp 16
  network 1.1.1.31/32 route-policy SID(31)
  allocate-label all
 !
 neighbor-group TIER2
  address-family ipv4 labeled-unicast
   route-reflector-client
   route-policy ADDMETRIC(10) out
   next-hop-self
  !
 !
 neighbor 99.11.31.11
  remote-as 1
  use neighbor-group TIER2
  description iBGP peer xrvr-11
 !
 neighbor 99.12.31.12
  remote-as 1
  use neighbor-group TIER2
  description iBGP peer xrvr-12
 !
 neighbor 99.21.31.21
  remote-as 1
  use neighbor-group TIER2
  description iBGP peer xrvr-21
 !
 neighbor 99.22.31.22
  remote-as 1
  use neighbor-group TIER2
  description iBGP peer xrvr-22
 !
!
```

The configurations of the other nodes are similar to the above configurations. The BGP entry for prefix 1.1.1.3/32 on Node1 is shown in Example 6-45. Two paths are available, one via Node11 (lines 10 to 20), and one via Node12 (lines 21 to 31). BGP has selected the first path as bestpath since it has the lowest neighbor address, and the other path is selected for BGP multipath. The AIGP metric for both paths is 40. The cluster-list of the first path is `1.1.1.11, 1.1.1.31, 1.1.1.21` which indicates it has been reflected by Node21, Node31, and Node11.

Example 6-45: BGP table entry of prefix 1.1.1.3/32 on Node1

```
 1 RP/0/0/CPU0:xrvr-1#show bgp ipv4 labeled-unicast 1.1.1.3/32
 2 BGP routing table entry for 1.1.1.3/32
 3 Versions:
 4   Process           bRIB/RIB  SendTblVer
 5   Speaker                123         123
 6     Local Label: 16003
 7 Last Modified: Aug 16 14:43:23.342 for 00:30:22
 8 Paths: (2 available, best #1)
 9   Not advertised to any peer
10   Path #1: Received by speaker 0
11   Not advertised to any peer
12   Local
13     99.1.11.11 from 99.1.11.11 (1.1.1.3)
14       Received Label 16003
15       Origin IGP, metric 0, localpref 100, aigp metric 40,
valid, internal, best, group-best, multipath
16       Received Path ID 0, Local Path ID 0, version 123
17       Originator: 1.1.1.3, Cluster list: 1.1.1.11, 1.1.1.31,
1.1.1.21
18       Total AIGP metric 40
19       Prefix SID Attribute Size: 10
20       Label Index: 3
21   Path #2: Received by speaker 0
22   Not advertised to any peer
23   Local
24     99.1.12.12 from 99.1.12.12 (1.1.1.3)
25       Received Label 16003
26       Origin IGP, metric 0, localpref 100, aigp metric 40,
valid, internal, multipath
```

```
      Received Path ID 0, Local Path ID 0, version 0
      Originator: 1.1.1.3, Cluster list: 1.1.1.12, 1.1.1.31,
1.1.1.21
      Total AIGP metric 40
      Prefix SID Attribute Size: 10
      Label Index: 3
```

Both paths are installed in RIB (see Example 6-46) and FIB (see Example 6-49) on Node1.

Example 6-46: RIB entry of prefix 1.1.1.3/32 on Node1

```
RP/0/0/CPU0:xrvr-1#show route 1.1.1.3/32

Routing entry for 1.1.1.3/32
  Known via "bgp 1", distance 200, metric 40, [ei]-bgp, labeled
unicast (3107) (AIGP metric), labeled SR, type internal
  Installed Aug 16 14:43:23.802 for 00:36:38
  Routing Descriptor Blocks
    99.1.11.11, from 99.1.11.11, BGP multi path
      Route metric is 40
    99.1.12.12, from 99.1.12.12, BGP multi path
      Route metric is 40
  No advertising protos.
```

Example 6-47: CEF entry of prefix 1.1.1.3/32 on Node1

```
RP/0/0/CPU0:xrvr-1#show cef 1.1.1.3/32
1.1.1.3/32, version 1270, internal 0x5000001 0x80 (ptr
0xa141b1f4) [1], 0x0 (0xa13e5c44), 0xa08 (0xa1758260)
 Updated Aug 16 14:43:23.822
 Prefix Len 32, traffic index 0, precedence n/a, priority 4
   via 99.1.11.11/32, 5 dependencies, recursive, bgp-multipath
[flags 0x6080]
    path-idx 0 NHID 0x0 [0xa15eb6f4 0x0]
    recursion-via-/32
    next hop 99.1.11.11/32 via 24000/0/21
     local label 16003
     next hop 99.1.11.11/32 Gi0/0/0/0    labels imposed
{ImplNull 16003}
   via 99.1.12.12/32, 5 dependencies, recursive, bgp-multipath
[flags 0x6080]
```

```
        path-idx 1 NHID 0x0 [0xa15eb7f4 0x0]
        recursion-via-/32
        next hop 99.1.12.12/32 via 24007/0/21
         local label 16003
         next hop 99.1.12.12/32 Gi0/0/0/1    labels imposed
{ImplNull 16003}
```

The service overlay configuration is very similar to the configuration for the EBGP case. The L3VPN service configuration of the Tier-1 node Node3 is shown in Example 6-48. IBGP multi-path is also enabled on the service overlay; see line 27. Both Node3 and Node4 advertise the vrf SVC prefix 99.3.9.0/24.

Example 6-48: BGP configuration for L3VPN service on Node3

```
 1 vrf SVC
 2  address-family ipv4 unicast
 3   import route-target
 4    1:1000
 5   !
 6   export route-target
 7    1:1000
 8   !
 9  !
10 !
11 router bgp 1
12  address-family vpnv4 unicast
13  !
14  neighbor-group ROUTE_REFLECTORS
15   remote-as 1
16   update-source Loopback0
17   address-family vpnv4 unicast
18   !
19  !
20  neighbor 1.1.1.41
21   use neighbor-group ROUTE_REFLECTORS
22   description service RR Node41
23  !
24  vrf SVC
25   rd auto
26   address-family ipv4 unicast
```

```
   maximum-paths ibgp 16
   network 99.3.9.0/24
  !
 !
!
```

Example 6-47 shows the BGP configuration for the service overlay on the Router Reflector Node41. The configuration for the underlay connectivity is not shown. The neighbor-group `RR_CLIENTS` on line 5 is for the L3VPN service neighbors, using `address-family vpnv4 unicast`. These L3VPN service neighbors are configured as Route Reflector clients.

Example 6-49: BGP configuration for L3VPN service on Service RR Node41

```
 1 router bgp 1
 2  bgp router-id 1.1.1.41
 3  address-family vpnv4 unicast
 4  !
 5  neighbor-group RR_CLIENTS
 6   remote-as 1
 7   update-source Loopback0
 8   address-family vpnv4 unicast
 9    route-reflector-client
10   !
11  !
12  neighbor 1.1.1.1
13   use neighbor-group RR_CLIENTS
14  !
15  neighbor 1.1.1.2
16   use neighbor-group RR_CLIENTS
17  !
18  neighbor 1.1.1.3
19   use neighbor-group RR_CLIENTS
20  !
21  neighbor 1.1.1.4
22   use neighbor-group RR_CLIENTS
23  !
24 !
```

The vpnv4 BGP prefix 99.3.9.0/24 in vrf SRV on Node1 is shown in Example 6-50. Both Node3 and Node4 advertise this prefix, therefore Node1 has two paths towards that prefix: one via Node3 (nexthop 1.1.1.3) and one via Node4 (nexthop 1.1.1.4). The label 90039 (line 13) is the Aggregate label for vrf SVC as allocated by Node3 for this vrf. Node4 allocated label 90049 (line 23) for this prefix.

Example 6-50: BGP table entry of multi-homed prefix 99.3.9.0/24 in vrf SVC on Node1

```
 1| RP/0/0/CPU0:xrvr-1#show bgp vrf SVC 99.3.9.0/24
 2| BGP routing table entry for 99.3.9.0/24, Route
Distinguisher: 1.1.1.1:0
 3| Versions:
 4|   Process           bRIB/RIB  SendTblVer
 5|   Speaker                  6           6
 6| Last Modified: Aug 17 12:09:06.342 for 00:00:05
 7| Paths: (2 available, best #1)
 8|   Not advertised to any peer
 9|   Path #1: Received by speaker 0
10|   Not advertised to any peer
11|   Local
12|     1.1.1.3 (metric 40) from 1.1.1.41 (1.1.1.3)
13|       Received Label 90039
14|       Origin IGP, metric 0, localpref 100, valid, internal,
best, group-best, multipath, import-candidate, imported
15|       Received Path ID 0, Local Path ID 0, version 6
16|       Extended community: RT:1:1000
17|       Originator: 1.1.1.3, Cluster list: 1.1.1.41
18|       Source AFI: VPNv4 Unicast, Source VRF: default, Source
Route Distinguisher: 1.1.1.3:0
19|   Path #2: Received by speaker 0
20|   Not advertised to any peer
21|   Local
22|     1.1.1.4 (metric 40) from 1.1.1.41 (1.1.1.4)
23|       Received Label 90049
24|       Origin IGP, metric 0, localpref 100, valid, internal,
multipath, import-candidate, imported
25|       Received Path ID 0, Local Path ID 0, version 0
26|       Extended community: RT:1:1000
27|       Originator: 1.1.1.4, Cluster list: 1.1.1.41
28|       Source AFI: VPNv4 Unicast, Source VRF: default, Source
```

```
Route Distinguisher: 1.1.1.4:0
```

Example 6-50 shows the vrf SVC RIB entry for prefix 99.3.9.0/24. Two paths are available: via Node3 and via Node4.

Example 6-51: RIB entry of multi-homed prefix 99.3.9.0/24 in vrf SVC on Node1

```
RP/0/0/CPU0:xrvr-1#show route vrf SVC 99.3.9.0/24

Routing entry for 99.3.9.0/24
  Known via "bgp 1", distance 200, metric 0, type internal
  Installed Aug 17 12:09:06.207 for 00:00:51
  Routing Descriptor Blocks
    1.1.1.3, from 1.1.1.41, BGP multi path
      Nexthop in Vrf: "default", Table: "default", IPv4
Unicast, Table Id: 0xe0000000
      Route metric is 0
    1.1.1.4, from 1.1.1.41, BGP multi path
      Nexthop in Vrf: "default", Table: "default", IPv4
Unicast, Table Id: 0xe0000000
      Route metric is 0
  No advertising protos.
```

6.9 Summary

- SR BGP Prefix-SIDs can be used for various deployment designs like end-to-end inter-AS (or inter-domain) MPLS connectivity, or BGP-based Data Centers.
- The BGP Labeled-Unicast mechanism (RFC 3107) has been extended for signaling of the BGP Prefix-SID
- SR BGP interoperates with non-SR BGP-LU implementations and can be enabled seamlessly in an existing network for migration to SR.
- SR BGP makes it easier to bring MPLS into Data Center and provides additional benefits.

6.10 References

- [draft-ietf-idr-bgp-prefix-sid] Previdi, S., Filsfils, C., Lindem, A., Patel, K., Sreekantiah, A., Ray, S., and H. Gredler, "Segment Routing Prefix SID extensions for BGP", draft-ietf-idr-bgp-prefix-sid (work in progress), June 2016, https://datatracker.ietf.org/doc/draft-ietf-idr-bgp-prefix-sid.
- [draft-ietf-spring-oam-usecase] Geib, R., Ed., Filsfils, C., Pignataro, C., Ed., and N. Kumar, "A Scalable and Topology-Aware MPLS Dataplane Monitoring System", Work in Progress, draft-ietf-spring-oam-usecase, September 2016, https://datatracker.ietf.org/doc/draft-ietf-spring-oam-usecase.
- [draft-ietf-spring-segment-routing-msdc] Filsfils, C., Previdi, S., Mitchell, J., Aries, E., and P. Lapukhov, "BGP-Prefix Segment in large-scale data centers", draft-ietf-spring-segment-routing-msdc-01 (work in progress), April 2016, https://datatracker.ietf.org/doc/draft-ietf-spring-segment-routing-msdc-01.
- [draft-ietf-spring-sr-oam-requirement] Kumar, N., Pignataro, C., Akiya, N., Geib, R., Mirsky, G., and S. Litkowski, "OAM Requirements for Segment Routing Network", Work in Progress, draft-ietf-spring-sr-oam-requirement, July 2016, https://datatracker.ietf.org/doc/draft-ietf-spring-sr-oam-requirement.
- [FLOWLET] Sinha, S., Kandula, S., and D. Katabi, "Harnessing TCP's Burstiness with Flowlet Switching", 2004.
- [RFC2283] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 2283, DOI 10.17487/RFC2283, February 1998, http://www.rfc-

editor.org/info/rfc2283, obsoluted by RFC4760.

- [RFC3107] Rekhter, Y. and E. Rosen, “Carrying Label Information in BGP-4”, RFC 3107, DOI 10.17487/RFC3107, May 2001, https://datatracker.ietf.org/doc/rfc3107.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., “A Border Gateway Protocol 4 (BGP-4)”, RFC 4271, DOI 10.17487/RFC4271, January 2006, https://datatracker.ietf.org/doc/rfc4271.
- [RFC4364] Rosen, E. and Y. Rekhter, “BGP/MPLS IP Virtual Private Networks (VPNs)”, RFC 4364, DOI 10.17487/RFC4364, February 2006, https://datatracker.ietf.org/doc/rfc4364.
- [RFC4456] Bates, T., Chen, E., and R. Chandra, “BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)”, RFC 4456, DOI 10.17487/RFC4456, April 2006, https://datatracker.ietf.org/doc/rfc4456.
- [RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, “Multiprotocol Extensions for BGP-4”, RFC 4760, DOI 10.17487/RFC4760, January 2007, https://datatracker.ietf.org/doc/rfc4760.
- [RFC7311] Mohapatra, P., Fernando, R., Rosen, E., and J. Uttaro, “The Accumulated IGP Metric Attribute for BGP”, RFC 7311, DOI 10.17487/RFC7311, August 2014, https://datatracker.ietf.org/doc/rfc7311.
- [RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., “Use of BGP for Routing in Large-Scale Data Centers”, RFC 7938, DOI 10.17487/RFC7938, August 2016, https://datatracker.ietf.org/doc/rfc7938.

[1] In a non-blocking fabric there is always a full rate path available between two servers, no matter what other paths exist at the time.

7 SEGMENT ROUTING IN EXISTING MPLS NETWORKS

One of the goals of Segment Routing has been to simplify the required protocols and mechanism required to operate existing MPLS while adding new protection and Traffic Engineering capabilities. MPLS is widely deployed today and as a result, introduction of Segment Routing technology in existing networks must be done in a simple and seamless manner without disruption of services. This has always been a critical requirement for people involved in the Segment Routing technology development from the very beginning. For the most part, the base Segment Routing features can be enabled with just a software upgrade since the MPLS forwarding plane is leverage as seen previously in chapter 7, “Segment Routing in Existing MPLS Networks”.

In this chapter we look at the deployment options and mechanisms available for introduction of Segment Routing in existing MPLS networks. To begin with, we look at the high-level deployment models. These models can be mixed, either just during a transition period or on a more permanent basis.

In the first approach Segment Routing and other MPLS protocols co-exist next to each other as “ships in the night”. Segment Routing can be introduced without disrupting existing services. New services can be deployed using Segment Routing while leaving the existing services on their existing transport. These existing services can already benefit from the Segment Routing functionality, such as Topology Independent LFA

Fast Reroute protection (we will see more details on this in chapter 9, "Topology Independent LFA (TI-LFA)"). Existing services can then be gradually migrated to using Segment Routing transport, ultimately migrating to a Segment Routing only network. Some services could be retained on their existing transport, if required, on a more permanent basis. This migration approach is driven primarily based on the consideration of services and their transports – assuming all routers provide flexibility to support existing as well as newer Segment Routing transport.

In the second approach, Segment Routing and LDP interwork with each other. This interworking functionality allows gradually upgrading and enabling Segment Routing on routers for the largest part of the network while leaving legacy equipment (without Segment Routing software support) in place, running LDP. The major part of the network can benefit from the Segment Routing capabilities while allowing time for replacement of legacy devices in production. Note that for leveraging some of the newer and advanced applications of Segment Routing like Traffic Engineering (which we shall see in the next part of this book), it may be required to also perform hardware upgrades on specific routers in the network. This migration approach is driven by the consideration of the router platforms deployed in the network, their positions or functions and their capabilities.

Different deployment models are possible by combining both of these approaches. Segment Routing provides the flexibility to introduce it in any of these different models based on the mix of services, transports, and routing platform considerations.

HIGHLIGHT: Segment Routing Deployment in

Existing MPLS networks

Segment Routing base features can be enabled via software upgrade on most routing platforms; hardware upgrades may be required only on certain advanced applications and that too on specific routers in the network.

Segment Routing transport can be introduced in a network for certain new services or a subset of existing services while other existing services continue to operate using their existing transport mechanisms; ships-in-the-night model.

Segment Routing features can be enabled in a network which has legacy equipment that does not support Segment Routing; LDP interworking allows for use of Segment Routing for services in such networks.

In the rest of this chapter, we will look closer at how Segment Routing co-exists and interworks with traditional MPLS transport technologies.

7.1 Co-Existence of SR and other MPLS Protocols

7.1.1 Control Plane Co-Existence

Multiple MPLS label distribution protocols can concurrently run on a node. They can co-exist as *ships-in-the-night* and their control plane mechanisms are independent. The MPLS architecture permits concurrent use of LDP, RSVP-TE, Segment Routing and others. This is nothing new; MPLS control plane co-existence exists since the beginning of MPLS.

An MPLS control plane function ("Label Manager") exists on each node that ensures that the local labels used by different label distribution protocols don't collide; it ensures that local labels are uniquely allocated and assigned.

For Segment Routing Global Segments, such as the Prefix Segments, a range of local labels is reserved: the SRGB. The Label Manager of each node ensures that labels in this range are exclusively used for these SR Global Segments.

For the MPLS label distribution protocols (e.g. LDP, RSVP-TE, BGP, etc.) that use random dynamic labels, or for the Segment Routing Local Segments, such as the IGP Adjacency Segments or BGP Peer Segments, the Label Manager ensures that each local label is locally unique, that each local label is allocated and assigned only once.

The Label Manager on each node can only locally manage the label assignment. For the Global Segments, it is the operator's responsibility to ensure that each Global Segment gets a SID index that is unique within the

Segment Routing Domain. Each global unique SID index then translates into a unique local label for that Global Segment on each node.

HIGHLIGHT: MPLS control plane protocols' co-existence

MPLS architecture has always allowed for multiple protocols like LDP, RSVP-TE, BGP and others to operate independently; the same still applies for Segment Routing control plane protocols like ISIS, OSPF and BGP.

The Label Manager ensures that signaled dynamic labels do not collide; SRGB ensures that the label space for SR Global Segments is reserved exclusively for it.

7.1.2 Data Plane Co-Existence

Multiple MPLS label distribution protocols can co-exist on a node. The Label Manager function takes care of the uniqueness of locally allocated labels. This unique local label assignment ensures that label switched paths installed by different MPLS control plane protocols can co-exist. Two cases can be distinguished: (1) the incoming packet is a labeled packet (MPLS-to-MPLS or MPLS-to-IP), and (2) the incoming label is an unlabeled packet (IP-to-MPLS). Even though the references here are for IPv4, the same also applies for IPv6.

7.1.2.1 MPLS-to-MPLS, MPLS-to-IP

On the data plane, multiple different MPLS-to-MPLS or MPLS-to-IP forwarding entries can co-exist, even if they lead towards the same destination prefix. As long as the local or incoming label values are locally unique, there is no conflict and no interference.

A packet arrives with a specific label on top of its label stack. That label has a unique entry in the MPLS forwarding table of the receiving node and forwarding is done based on that unique entry. The Label Manager ensures the uniqueness of each local label.

The outgoing labels of MPLS-to-MPLS forwarding entries don't need to be unique, since the outgoing labels are only significant for the downstream neighbor, the neighbor that the packet is forwarded to, not for the local node.

Multiple different MPLS-to-MPLS and MPLS-to-IP forwarding entries can co-exist for the same destination – e.g. in the case where there is a Prefix Segment as well as an LDP LSP towards the same destination prefix. Even though they are associated with the same destination prefix on the control plane, each control plane protocol independently programs its entry as a label cross-connect ("label in to label out") entry, each with its unique local label.

Figure 7-1 illustrates the data plane co-existence of MPLS forwarding entries installed by Segment Routing and MPLS entries installed by other MPLS label distribution protocols. The other MPLS protocol is LDP in this example, but it can be any other MPLS protocol such as RSVP-TE, or BGP-LU, etc.

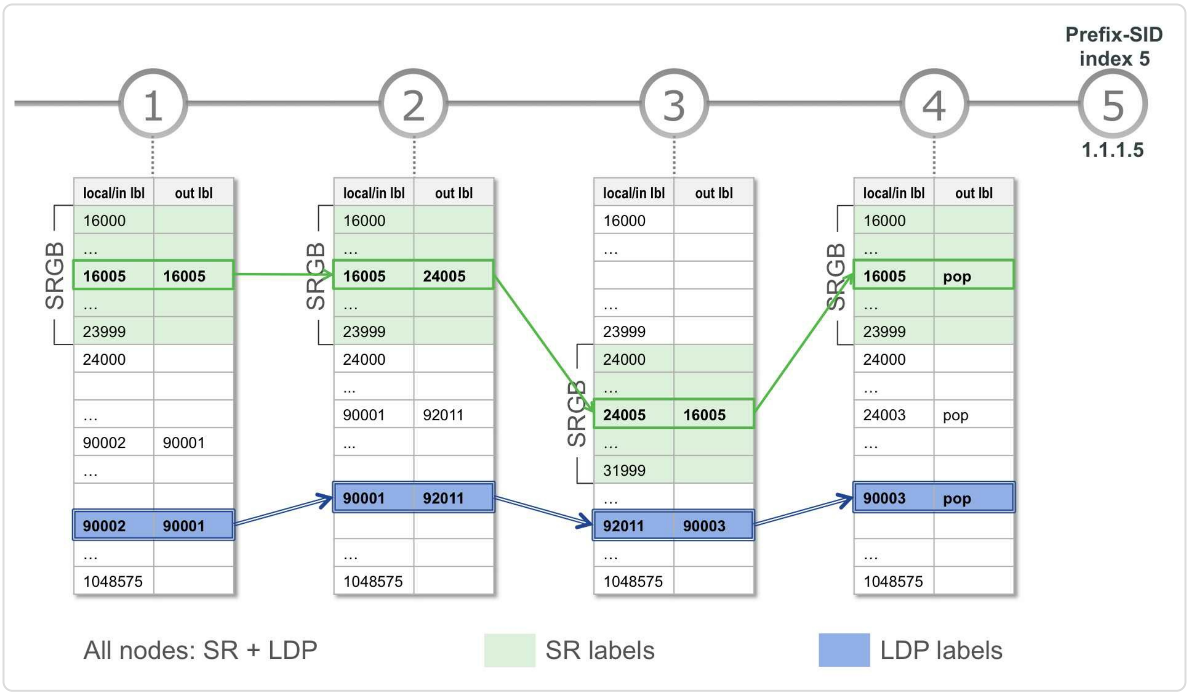


Figure 7-1: MPLS-to-MPLS, MPLS-to-IP co-existence

The network topology is a chain of nodes. The path through the nodes will be followed from Node1 to Node5. The MPLS forwarding table of each node is displayed under the node, with the incoming or local label in the left column, and the outgoing label in the right column. The label table is sorted with incrementing local label values. The usable label range in Cisco IOS XR for these protocols is from 16000 to 1 million. The labels smaller than 16000 are reserved in Cisco IOS XR for static label entries and special-purpose labels. All nodes in this topology have both Segment Routing and LDP enabled.

Node1, Node2 and Node4 have allocated the default SRGB, from 16000 to 23999. Although the recommendation is to “use the same SRGB on all nodes”, Node3 has allocated a different SRGB for illustration purposes. The SRGB of Node3 starts at 24000 and goes up to 31999.

Node5 advertises its loopback prefix 1.1.1.5/32 with an associated Prefix-SID index 5. The default PHP behavior is requested for this Prefix-SID. On Node1, Node2, and Node4 this Prefix-SID has a local label value of 16005 (=16000 + 5). On Node3 the local label value of this prefix-SID is 24005 (=24000 + 5).

Figure 7-1 shows the label forwarding entries that are used when forwarding a packet on the Prefix Segment to Node5. When a packet enters Node1 with a top label 16005, it will be forwarded using the highlighted label forwarding entries until the label is popped at the penultimate hop Node4. Node5 then processes the packet based on the packet header that is exposed after Node4 popped the top label.

All nodes also allocate and advertise a LDP label for the loopback prefix of Node5, 1.1.1.5/32. Figure 7-1 shows the label forwarding entries that are used when forwarding a packet on the LDP Label Switched Path to the destination at Node5.

When a packet enters Node1, with a top label 90002, which is the local LDP label allocated by Node1 for the loopback prefix of Node5, it is forwarded with the highlighted label forwarding entries until the label is popped at the penultimate hop Node4. Node5 then processes the packet based on the newly exposed header.

Since the local Label Manager on each node manages the local label allocation, it is guaranteed that these local labels are unique. Since their local or incoming labels catalog the MPLS-to-MPLS and MPLS-to-IP forwarding entries, these MPLS forwarding entries can co-exist, regardless of the MPLS applications that installed each forwarding entry. Packets transported by MPLS forwarding entries installed by different MPLS label distribution protocols are as ships-in-the-night.

The example showed co-existing Segment Routing and LDP label forwarding entries, but the MPLS control plane and data plane co-existence also applies to MPLS labels distributed by any other MPLS label distribution protocols such as RSVP-TE and BGP Labeled Unicast.

7.1.2.2 IP-to-MPLS

When an IP packet arrives on the ingress node of an MPLS network, the packet is classified into a Forwarding Equivalence Class. A label is then imposed on the packet, depending on the packet's FEC. By default the classification of packets into a FEC is based on a longest prefix match of the packet's destination address. This default classification of packets and label imposition on those packets is the subject of this section. More granular packet classification or classification on other elements than the destination address are possible, but this is outside of the scope of this section and those classification rules are not really impacted or changed by introduction of Segment Routing.

When an IP packet arrives on the ingress node of the MPLS network, a longest matching prefix lookup for the destination address is done in the forwarding table, the Forwarding Information Base (FIB). This lookup results in a single FIB entry, the prefix that is the longest matching prefix in the FIB for the packet's destination address. Each FIB entry has one or multiple equal cost paths leading to the destination. If multiple equal cost paths exist for a prefix then the traffic destined to that prefix is load-balanced over the available paths. Each of these prefix paths has a single outgoing label entry that contains the label that is imposed on packets forwarded on that path. Normally a single label would be imposed per prefix path.

In a mixed Segment Routing and LDP environment where a destination can be reached via a Prefix Segment as well as via a LDP LSP, only one label imposition entry can be programmed for the destination prefix path. Either a Segment Routing label or a LDP label can be imposed on packets forwarded on that prefix path. If multiple equal cost paths exist to the destination, each of the individual prefix paths can either have a Segment Routing or an LDP label imposition entry, independent from the other prefix paths of the same prefix. First we take a look at the usual and simpler scenario where the path(s) is (are) from a single protocol and further in section 7.2, “Segment Routing and LDP Interworking” we will look at a mix of paths when we look at interworking.

In Figure 7-2, which is based on Figure 7-1, two labeled paths exist from Node1 to destination Node5. The first path is a Segment Routing installed path: the Prefix Segment to Node5. The other path is an LDP installed path: the LDP LSP to Node5. If an unlabeled packet with destination address 1.1.1.5 arrives on Node1, which label should be imposed on the packet to transport it to its destination?

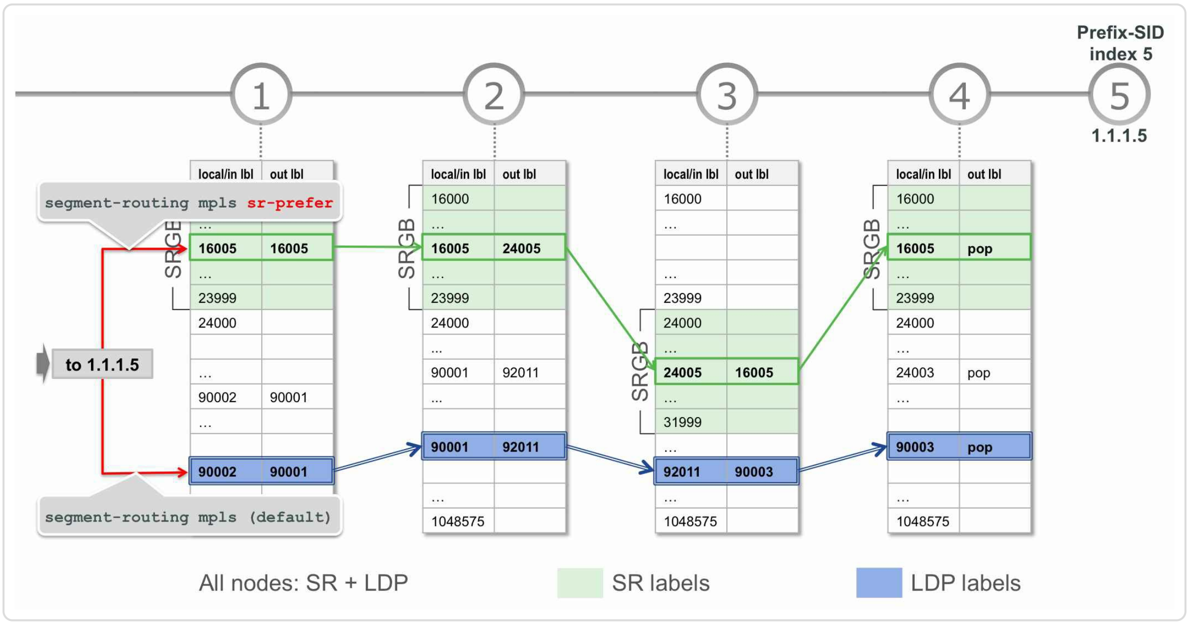


Figure 7-2: no IP-to-MPLS co-existence: select label to impose

By default, Node1 imposes the LDP label for the FEC 1.1.1.5/32, which is 90001 in the example. But the operator can configure Node1 to prefer the Segment Routing label imposition instead. In the example that would be label 16005, the Prefix-SID associated with 1.1.1.5/32.

In the case where only a single labeled path is available towards the destination (either Segment Routing or LDP, but not both), then there is no choice, the label of the only available labeled path will be imposed on the incoming unlabeled packet.

The decision to prefer LDP rather than Segment Routing label imposition by default has been made after much thought. The idea is that, as Segment Routing is new at the time of writing this book, it is very likely that networks use LDP for their transport signaling. When Segment Routing is enabled in a rolling manner across nodes in the network, we do not necessarily want it to be used immediately by default; all services continue to operate without interruption using LDP. This gives an opportunity for operators to check and verify the Segment Routing control plane operations and the forwarding entries as well without really shifting services over it. Then, at a suitable time, the operator can change the preference to seamlessly switch over the services from LDP to Segment Routing. This switch can be done one ingress or edge node at a time and even for specific services or destinations only. Perhaps some time in the not so distant future, as Segment Routing deployment and usage becomes more pervasive, it would be natural to change the default preference to be Segment Routing.

On Cisco IOS-XR platforms, if an unlabeled packet arrives on the ingress node of an MPLS network and there is an LDP outgoing label as well as a Segment Routing outgoing label available for the packet's destination address, one of the two must be selected to impose on the packet. By default the LDP label imposition is preferred, so if an LDP outgoing label is available for the FEC bound to the destination prefix, that LDP label will be imposed on the packet. The operator can configure the ingress node to prefer imposing the Segment Routing label on the packet instead.

Adding the `sr-prefer` keyword to the `segment-routing mpls` configuration command under the IGP, configures the preference of Segment Routing over LDP label imposition. See Example 7-1 for ISIS and Example 7-2 for OSPF.

Example 7-1: SR-prefer configuration for ISIS

```
router isis 1
 address-family ipv4|ipv6 unicast
  segment-routing mpls sr-prefer
```

Example 7-2: SR-prefer configuration for OSPF

```
router ospf 1
```

```
  segment-routing mpls
  segment-routing mpls sr-prefer
```

The OSPF implementation also allows for the use of a prefix-list to determine specific prefixes that need to be preferred for Segment Routing paths. The prefix-list can be specified with the `sr-prefer` configuration. With this configuration SR label imposition is only preferred for the prefixes that are permitted by the prefix-list. This functionality allows to a granular migration from LDP to SR, by incrementally adding prefixes to the prefix-list. This gives a per-prefix control of the migration process. At the time of writing this book, the prefix-list option was not available with ISIS.

In the configuration shown in Example 7-3, SR label imposition is preferred for prefix 1.1.1.2/32 and all /32 prefixes in prefix range 2.1.1.0/24. LDP label imposition is preferred for all other prefixes.

Example 7-3: OSPF SR preference with prefix-list

```
ipv4 prefix-list PREFER_SR
 10 permit 1.1.1.2/32
 20 permit 2.1.1.0/24 eq 32
!
router ospf 1
 segment-routing sr-prefer prefix-list PREFER_SR
```

HIGHLIGHT: MPLS data plane co-existence

Multiple MPLS-to-MPLS and MPLS-to-IP forwarding entries can co-exist for the same IP Prefix as long as their incoming/local label is different.

The Label Manager ensures the uniqueness of these incoming/local labels either by directly controlling the unique allocation to local MPLS clients (LDP, RSVP-TE, BGP-VPN, etc) or indirectly by delegating a range of local labels (SRGB) to SR. SR itself ensures that the Prefix SID allocation within the SRGB is unique.

For IP-to-MPLS forwarding entries, the default preference is for LDP over Segment Routing; this can be changed via configuration to prefer Segment Routing

Considerations when Migrating to SR

“When enabling Segment Routing in an existing MPLS network, it is important to plan the roadmap for migration and the eventual design including aspects like:

- The SRGB size, the SR domain boundaries and design
- The router capabilities - which ones would support Segment Routing and which ones not; whether any upgrades required, etc.
- The services migrations - which ones would be migrated and when and which ones not

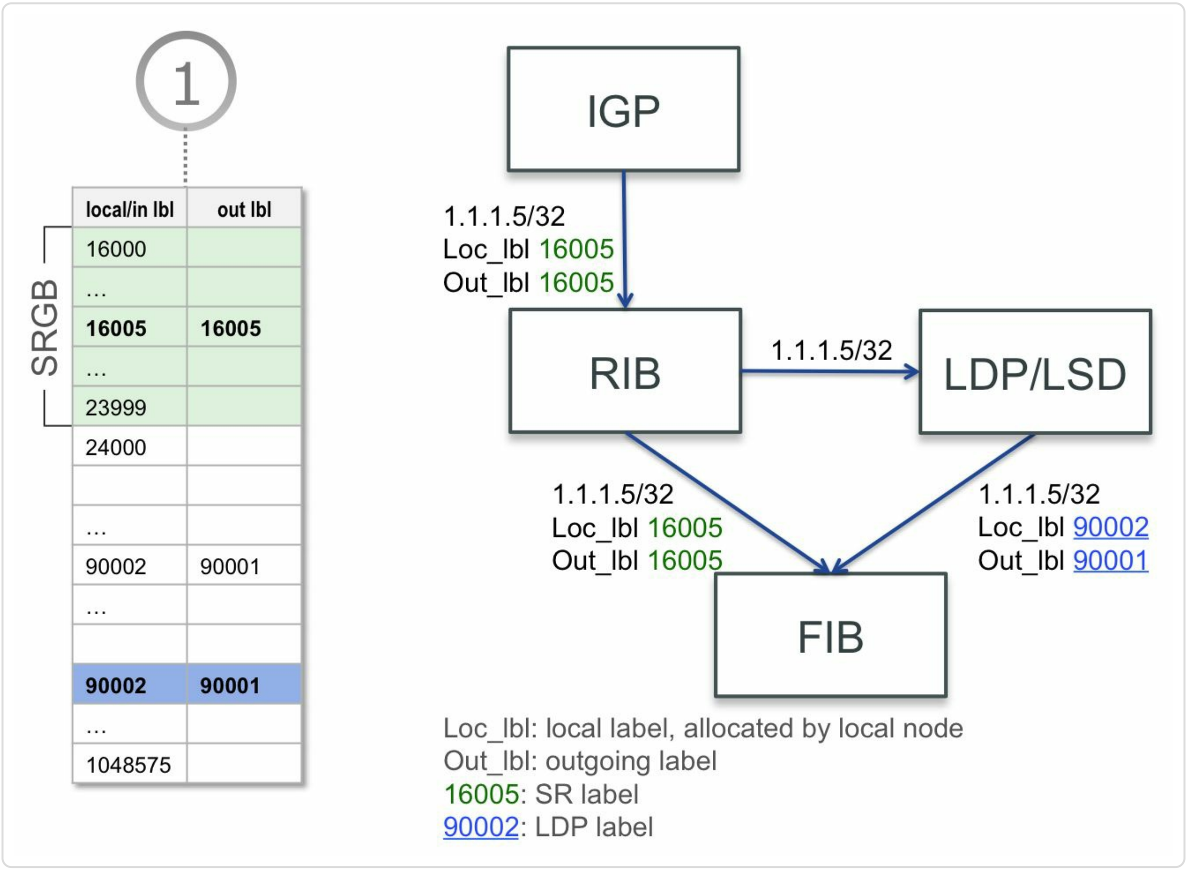
Once this is concluded then the following sequence is suggested to ensure a transition with minimal (if any) disruption to existing services:”

- Enable Segment Routing protocols in the control plane but do not set preference for Segment Routing in the forwarding plane
- Start provisioning of Prefix SIDs and SRMS ranges
- Verify the control plane operations and Segment Routing forwarding entries; also verify via OAM operations; check specifically the forwarding entries at SR/LDP interworking points
- If existing network had protection, then consider enabling TI-LFA as next step or else do this after the forwarding has been verified in the following step
- Start turning on preference for SR over LDP on certain nodes in the network and verify the seamless transition from LDP to SR; then roll over the same across the network
- At any step one can roll back to the previous state in case of issues as well

— Ketan Talaulikar

7.1.3 Implementation Aspects

To understand how the Segment Routing and LDP forwarding entries end up in the forwarding table, it is better to look at a highly simplified diagram of interactions between the different components involved taking Cisco IOS-XR implementation as a reference. The diagram in Figure 7-3 illustrates the behavior of Node1 in the example topology we have seen in the previous section (Figure 7-1 and Figure 7-2). The diagram shows the Interior Gateway Protocol (IGP) that programs its routes in the Routing Information Base (RIB). RIB programs the forwarding entries in the Forwarding Information Base (FIB), and forwards the prefix information to the Label Distribution Protocol (LDP) and Label Switching Database (LSD) components, which also program the forwarding entries in the FIB.



local/in lbl	out lbl
16000	
…	
16005	**16005**
…	
23999	
24000	
…	
90002	90001
…	
90002	**90001**
…	
1048575	

Figure 7-3: IGP and LDP programming FIB: behavior of Node1

IGP learns about a prefix 1.1.1.5/32 and its Prefix-SID index 5. In the topology there is a single path on Node1 towards that prefix, no ECMP. This simplifies the illustration, but remember that the described behavior applies to each individual prefix path. IGP installs its routes in RIB providing the prefix, the prefix paths and the prefix-SID labels of each path. In the example, the IGP installs prefix 1.1.1.5/32 in RIB. IGP provides the local label and outgoing label for the prefix. Since all nodes in the network use SRGB [16000-23999], both local and outgoing labels for Prefix-SID index 5 have value 16005 (= 16000 + 5). RIB forwards this prefix entry to LDP/LSD, which provides the local, and outgoing LDP labels for the FEC bound to this destination prefix path and sends the prefix path with the LDP labels to FIB. In parallel, RIB also directly sends the prefix path and the IGP provided Segment Routing labels to FIB.

Consequently, for the same prefix path, FIB receives information from two sources: RIB and LSD. It’s important to note that the initial source of both prefix path entries received by FIB is the IGP prefix path.

From these two sources, FIB prefers the LDP/LSD provided path by default. The operator can configure to prefer the Segment Routing (i.e. IGP) path instead.

For the preferred prefix path, FIB installs both a label imposition entry (to impose a label on incoming unlabeled packets with destination prefix 1.1.1.5/32 in this example) and a label cross-connect, which is a label swap entry or label disposition entry if this node is the penultimate hop and the label must be popped.

For the non-preferred prefix path, which is by default the IGP Segment Routing path, only the label cross-connect is installed. Or a label disposition entry if this node is the penultimate hop and the label must be popped.

Figure 7-4 zooms in on the FIB and illustrates the default case where the LDP label imposition is preferred. Three entries are programmed in FIB, all pertaining to the same prefix entry 1.1.1.5/32:

- impose label 90001 for IP packets matching prefix 1.1.1.5/32 (LDP)
- swap incoming label 90002 with outgoing label 90001 (LDP)
- swap incoming label 16005 with outgoing label 16005 (SR)

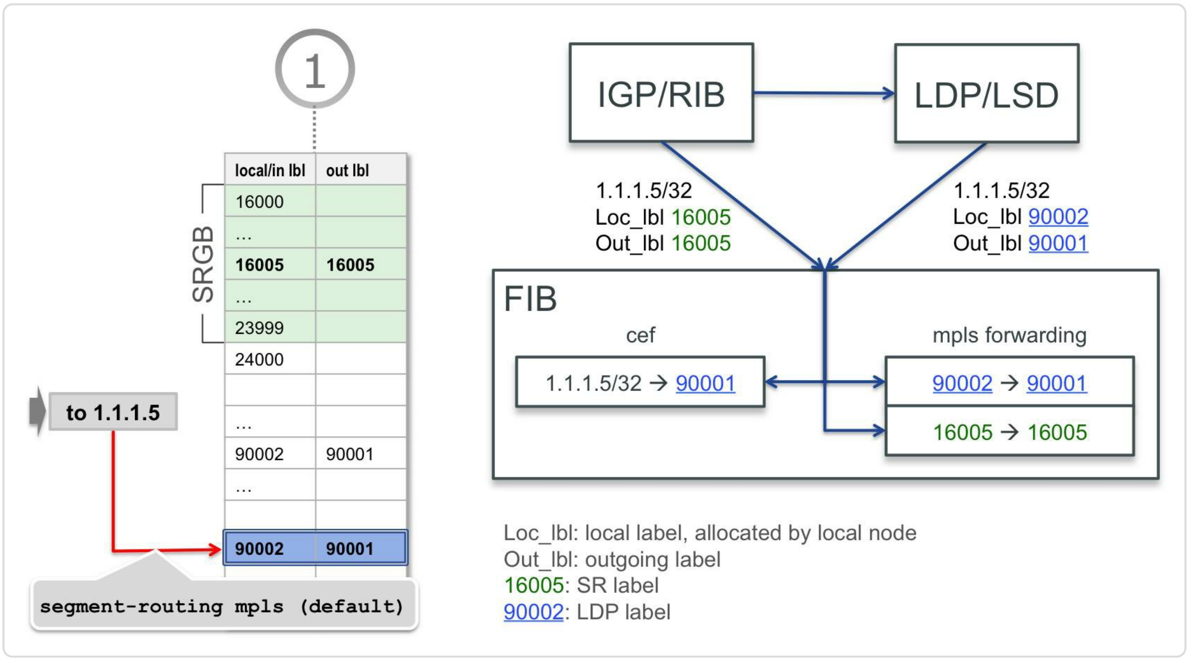


Figure 7-4: IGP and LDP programming FIB: prefer LDP label imposition

Assume now that `segment-routing mpls sr-prefer` is configured under the IGP configuration of Node1. The diagram in Figure 7-5 illustrates this non-default case, where Segment Routing label imposition is preferred. From the two sources that provide the prefix path forwarding information to FIB, RIB and LSD, FIB now prefers the IGP/RIB provided path.

For the preferred prefix path, FIB installs a label imposition entry (to impose a label on incoming unlabeled packets with destination prefix 1.1.1.5/32 in this example) and a label cross-connect.

For the non-preferred path, which is the LDP/LSD path in this case, only the label cross-connect is installed.

In the example of Figure 7-5, three entries are programmed in FIB, all concerning the same prefix entry 1.1.1.5/32:

- impose label 16005 for IP packets matching prefix 1.1.1.5/32 (SR)
- swap incoming label 16005 with outgoing label 16005 (SR)
- swap incoming label 90002 with outgoing label 90001 (LDP)

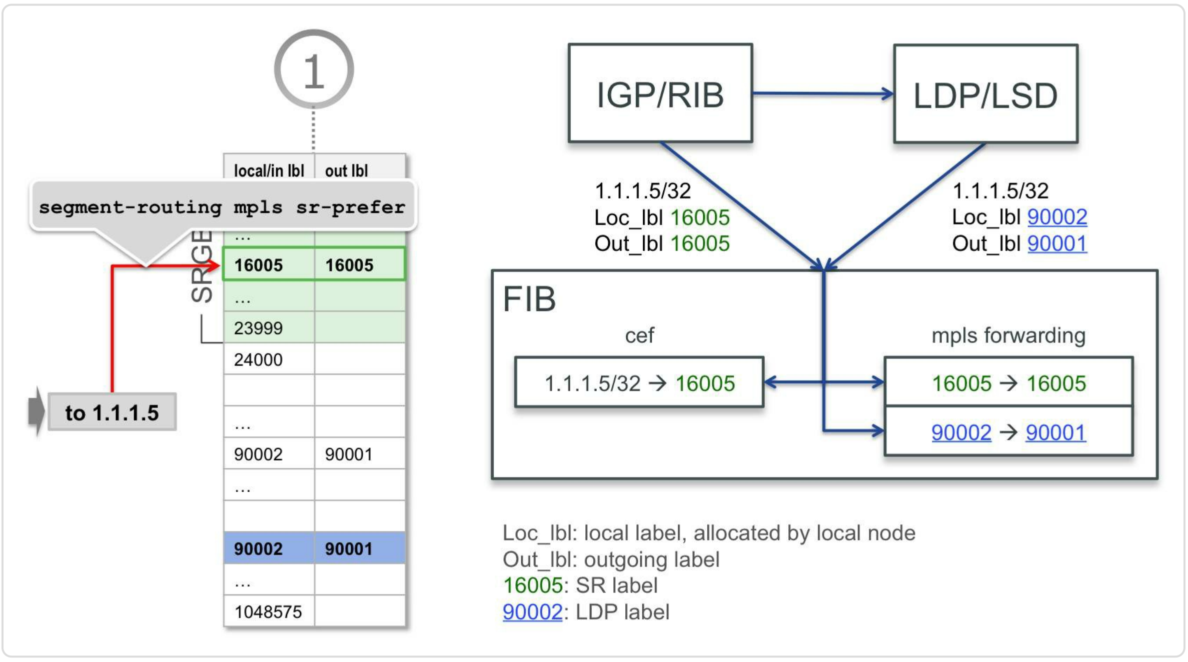


Figure 7-5: IGP and LDP programming FIB, prefer SR label imposition

Note that the label swap entries for both Segment Routing and LDP are always installed when available, regardless of the imposition preference setting. This is possible since multiple label swap forwarding entries for the same destination prefix can co-exist, there is no need to select between them.

The FIB programming is illustrated using the network topology in Figure 7-6. It is the same topology as shown in Figure 7-1. Node1 applies the default label imposition preference: impose the LDP outgoing label if it is available.

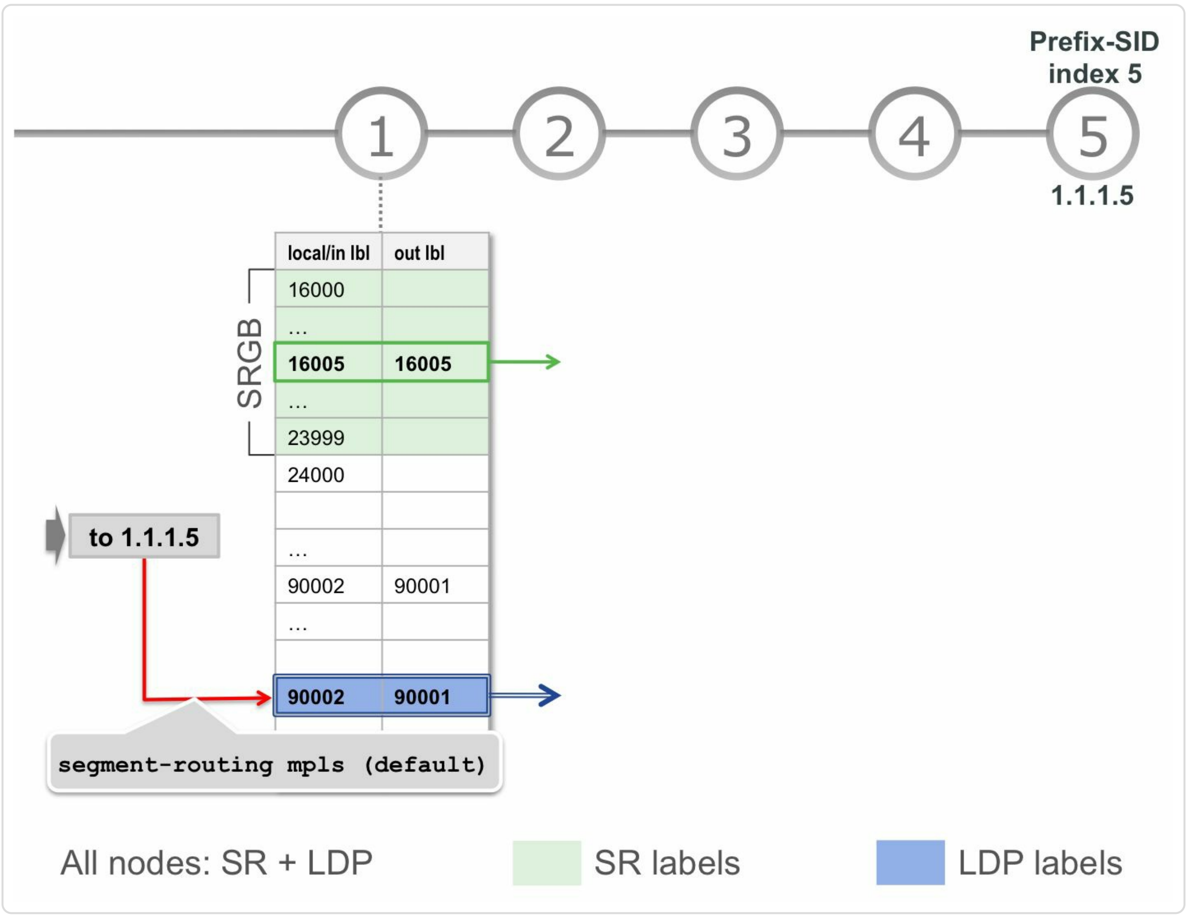


Figure 7-6: IGP and LDP programming FIB – example

Example 7-4 shows the output of `show mpls forwarding`, displaying the MPLS forwarding table entries on Node1 for the labels in this example. Both MPLS label cross-connects, one with Segment Routing labels (in/out 16005) and one with LDP labels (in 90002, out 90001), are programmed in the MPLS forwarding table of Node1. Both entries have the same outgoing interface and nexthop Node2. Note that both these forwarding entries are programmed regardless of the label imposition preference setting.

Example 7-4: IGP and LDP programming FIB – MPLS forwarding entries

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 16005
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16005  16005       SR Pfx (idx 5)     Gi0/0/0/0
99.1.2.2        0

RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 90002
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ------------
--- ------------
90002  90001       1.1.1.5/32         Gi0/0/0/0
99.1.2.2        0
```

Example 7-5 shows an extract of the ISIS configuration on Node1. The default LDP label imposition preference applies with this configuration.

Example 7-5: IGP and LDP programming FIB – ISIS configuration

```
router isis 1
 address-family ipv4 unicast
  metric-style wide
  segment-routing mpls
 !
!
```

The LDP local and outgoing label for the FEC bound to destination prefix 1.1.1.5/32 can be found in the `show mpls ldp bindings` output for prefix 1.1.1.5/32. See the output in Example 7-6. This command shows the LDP label binding for prefix 1.1.1.5/32 as advertised by the downstream neighbor Node2 (LDP router ID 1.1.1.2). Node2 allocated local label 90001 for prefix 1.1.1.5/32. The local LDP label allocated by Node1 for prefix 1.1.1.5/32 is 90002, and the outgoing label for this prefix is 90001.

Example 7-6: IGP and LDP programming FIB – LDP label bindings

```
RP/0/0/CPU0:xrvr-1#show mpls ldp bindings 1.1.1.5/32
1.1.1.5/32, rev 40
        Local binding: label: 90002
        Remote bindings: (1 peers)
            Peer                Label
            -----------------   ---------
            1.1.1.2:0           90001
```

These labels can then be found in the output of `show cef` for prefix 1.1.1.5/32 on Node1. See the output in Example 7-7. The CEF entry for prefix 1.1.1.5/32 shows the local label 90002 and the outgoing label, “labels imposed”, is 90001. These are indeed the LDP labels as found in the output of Example 7-6. According to this CEF entry, all packets destined to 1.1.1.5/32 and all packets for which the destination resolves on 1.1.1.5/32, such as BGP destinations with 1.1.1.5 as BGP nexthop, get outgoing label 90001 imposed and are transported over the LDP LSP.

Example 7-7: IGP and LDP programming FIB – CEF entry on Node1

```
RP/0/0/CPU0:xrvr-1#show cef 1.1.1.5/32
1.1.1.5/32, version 42, internal 0x1000001 0x1 (ptr 0xa13fa974)
[1], 0x0 (0xa13dfe3c), 0xa28 (0xa1541960)
 Updated Feb 22 13:42:56.375
 local adjacency 99.1.2.2
 Prefix Len 32, traffic index 0, precedence n/a, priority 3
```

```
   via 99.1.2.2/32, GigabitEthernet0/0/0/0, 11 dependencies,
weight 0, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0xa0ef31fc 0x0]
    next hop 99.1.2.2/32
    local adjacency
     local label 90002      labels imposed {90001}
```

In the next example the configuration of Node1 is updated to prefer Segment Routing labels for imposition. Therefore the operator added the `sr-prefer` keyword to the Segment Routing configuration of Node1. See the configuration extract in Example 7-8.

Example 7-8: IGP and LDP programming FIB – SR-prefer configuration on Node1

```
router isis 1
 address-family ipv4 unicast
  metric-style wide
  segment-routing mpls sr-prefer
 !
!
```

The IGP provides the prefix, prefix paths, and Segment Routing labels to RIB. RIB subsequently forwards the information to FIB. The output of `show route detail` for prefix 1.1.1.1/32 can be used to display the labels that IGP provided to RIB. See the output collected on Node1 in Example 7-9. The route to 1.1.1.5/32 has a local label 16005 and an outgoing label 16005 for the prefix path; both are highlighted in the output. Label 16005 is indeed the label of the Prefix-SID of 1.1.1.5/32.

Example 7-9: IGP and LDP programming FIB – RIB entry on Node1

```
RP/0/0/CPU0:xrvr-1#show route 1.1.1.5/32 detail

Routing entry for 1.1.1.5/32
  Known via "isis 1", distance 115, metric 40, labeled SR, type
level-2
  Installed Feb 22 13:56:14.630 for 00:00:03
```

```
    Routing Descriptor Blocks
      99.1.2.2, from 1.1.1.5, via GigabitEthernet0/0/0/0
        Route metric is 40
        Label: 0x3e85 (16005)
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        Path id:1       Path ref count:0
        NHID:0x1(Ref:19)
    Route version is 0x1d0 (464)
    Local Label: 0x3e85 (16005)
    IP Precedence: Not Set
    QoS Group ID: Not Set
    Flow-tag: Not Set
    Fwd-class: Not Set
    Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD
  Type RIB_SVD_TYPE_LOCAL
    Download Priority 1, Download Version 4267
    No advertising protos.
```

The `show cef` output for prefix 1.1.1.5/32 on Node1, as displayed in Example 7-10, shows that the prefix-SID label 16005 is imposed on packets destined for, or resolving on, prefix 1.1.1.5/32. The FIB preferred the IGP provided SR label information to program the forwarding entry.

Example 7-10: IGP and LDP programming FIB – CEF entry on Node1

```
  RP/0/0/CPU0:xrvr-1#show cef 1.1.1.5/32
  1.1.1.5/32, version 4267, internal 0x1000001 0x83 (ptr
  0xa13fa974) [1], 0x0 (0xa13dfb00), 0xa28 (0xa15413c0)
   Updated Dec 18 12:35:36.580
   local adjacency 99.1.2.2
   Prefix Len 32, traffic index 0, precedence n/a, priority 1
     via 99.1.2.2/32, GigabitEthernet0/0/0/0, 13 dependencies,
  weight 0, class 0 [flags 0x0]
      path-idx 0 NHID 0x0 [0xa0ef31fc 0x0]
      next hop 99.1.2.2/32
      local adjacency
       local label 16005      labels imposed {16005}
```

The examples in this section are using an ISIS network, but the equivalent functionality exists for OSPF. The examples are using IPv4 prefixes, but the same mechanisms are applicable to IPv6 prefixes if concurrently using other MPLS protocols (including LDP for IPv6) and Segment Routing MPLS for IPv6.

7.2 Segment Routing and LDP Interworking

In the interworking deployment model, the network needs to interwork Segment Routing parts of the network with LDP-only parts of the network. The interworking functionality takes care of Segment Routing to LDP and LDP to Segment Routing connectivity. It also takes care of interconnecting Segment Routing parts of the network over LDP and interconnecting LDP parts of the network using Segment Routing.

HIGHLIGHT: SR/LDP interworking

The SR/LDP interworking is seamless: no specific configuration is required (aside the Mapping Server (see chapter 8, "Segment Routing Mapping Server") for the LDP-only destinations), no specific gateway is defined. The interworking occurs automatically at any node on the border between the SR and LDP domains.

The seamless interworking is achieved by replacing an unknown outgoing label (Unlabelled) from one protocol by a valid outgoing label from the other protocol.

Avoiding the notion of a specific gateway is excellent as gateways represent congestion hot spots and availability risks.

SR nodes learn the SIDs of remote non-SR (LDP) nodes thanks to the Mapping Server advertisements.

Like for any SR node, the operator allocates a SID from the SRGB to each loopback prefix of an LDP and non-SR node. For example, if Node 1 with prefix 1.1.1.1/32 is running LDP but not SR, the operator would allocate index 1 of the SRGB to Prefix 1.1.1.1/32. This mapping (1.1.1.1/32, SID index 1) is then configured on the Mapping Server(s).

We advise the operator to activate the Mapping Server functionality on two nodes of the SR Domain (for redundancy purpose). The Mapping Servers advertise the (Prefix to SID) mappings on behalf of the LDP non-SR nodes. All other SR nodes of the domain are client by default.

7.2.1 LDP to SR Interworking

Figure 7-7 shows a network that is divided in two parts: an LDP part on the left side and an SR part on the right side. The nodes in the LDP part are only running LDP (no SR) and the nodes in the SR part are only running SR (no LDP). Node3, the node on the border between the two parts, runs both SR and LDP.

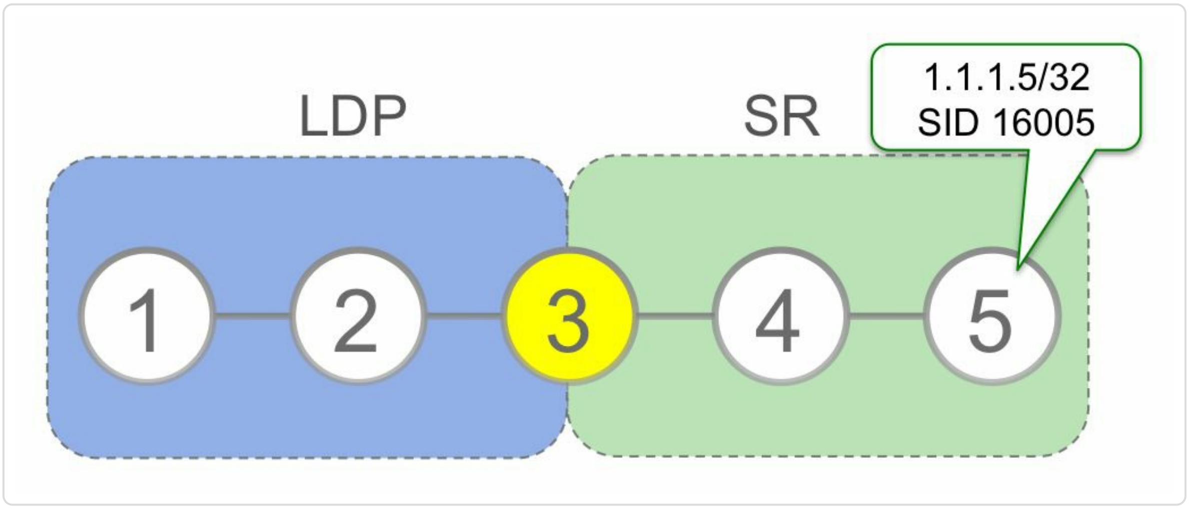


Figure 7-7: SR/LDP interworking network topology

How can the LDP-only Node1 reach the Segment Routing-only Node5 over a label switched path?

Node3 is LDP enabled, but its downstream neighbor Node4 towards the destination Node5, is not LDP enabled and consequently does not advertise any LDP labels. Therefore Node3 does not receive an LDP outgoing label for destination Node5. Note that if Node3 uses LDP Independent Label Distribution Control Mode then Node3 by default allocates an LDP local label for 1.1.1.5/32, since that prefix is in its routing table. If Node3 uses LDP Ordered Label Distribution Control Mode then Node3 must ensure that it allocates a local LDP label for remote prefixes with an associated Prefix-SID, and distributes that label to its downstream neighbors.

REMINDER: Label Distribution Control Mode

LDP can allocate local labels and distribute label bindings in two modes:

Independent Label Distribution Control Mode: LDP allocates and distributes a local label for every IP prefix in the node's routing table. LDP does not wait for the downstream neighbor to advertise a label binding for the prefix first.

Ordered Label Distribution Control Mode: LDP allocates and distributes local labels for IP prefixes in the node's routing table for which it is the egress node (i.e. last MPLS node for that prefix) or if it has received a label binding for that prefix from its downstream neighbor.

Cisco IOS XR routers always act in Independent Label Distribution Control Mode. By default Cisco IOS XR allocates an LDP local label for all prefixes in its routing table except BGP prefixes.

Normally, if Node3 does not receive an LDP outgoing label for 1.1.1.5/32, it installs an MPLS forwarding entry for 1.1.1.5/32 with an "Unlabelled" outgoing label. However, instead of programming an unlabeled entry in the forwarding table, Node3 automatically connects the LDP Label Switched Path towards Node5, to the Prefix Segment of Node5. Any node on the LDP to Segment Routing border automatically installs such LDP-to-SR forwarding entries.

To connect the LDP LSP to the Prefix Segment, Node3 installs the following LDP to Segment Routing MPLS forwarding entry:

- The local or incoming label is the local label allocated by LDP for the FEC of destination Node5.
- The outgoing label is the Prefix-SID label associated to Node5's

loopback prefix, 16005 in this example.

- The outgoing interface is the interface towards neighbor Node4, the downstream neighbor on the shortest path to Node5.

This interworking forwarding entry is automatically derived and installed by any LDP/SR border node. No additional configuration is required; no specific position in the network is required.

The interworking functionality only applies to labeled packets; no interworking functionality is needed for label imposition. For unlabeled IP packets with destination Node5 that arrive on Node3, Node3 imposes the Prefix-SID label 16005.

Figure 7-8 illustrates the MPLS forwarding table programming in a network with LDP to Segment Routing interworking. The figure shows the path from Node1 to destination Node5. The MPLS forwarding tables are shown under the nodes. The available label range is 16000 to 1 million. The left column of the forwarding tables shows the local or incoming labels, the right column shows the outgoing labels.

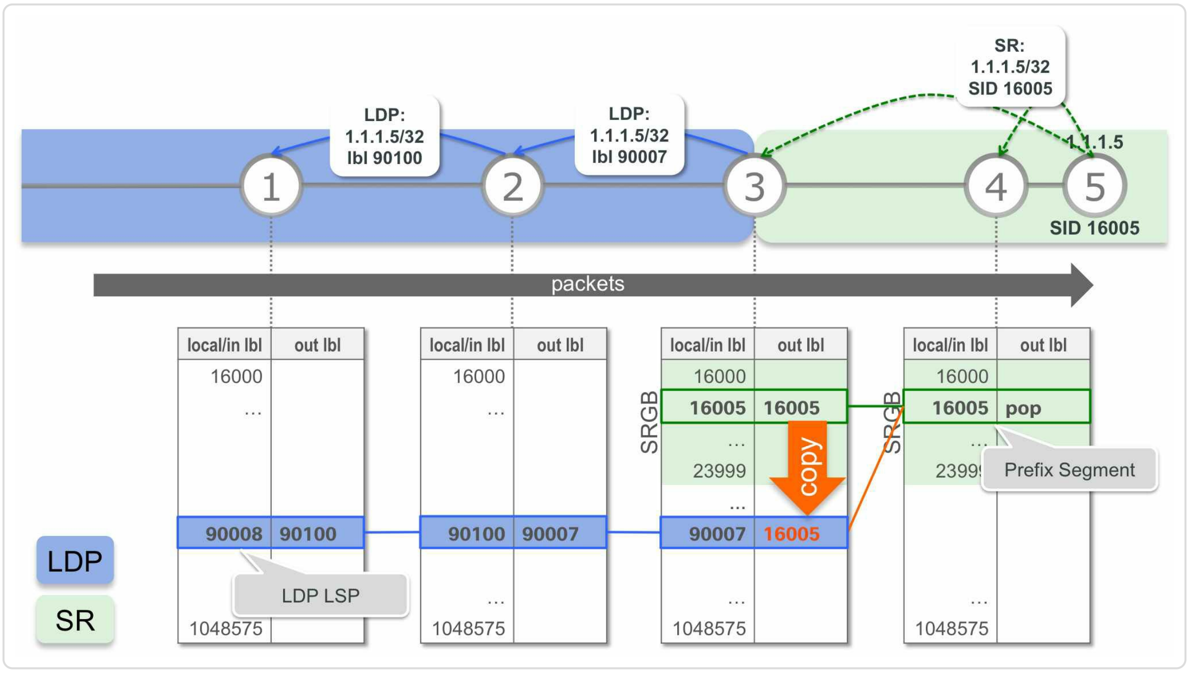


Figure 7-8: SR/LDP interworking illustration

Node1 and Node2 are legacy LDP-only nodes; they don't support Segment Routing. Thus Node1 and Node2 don't have an SRGB. It is assumed in this example that their dynamic label range starts at 24000.

Node4 and Node5 are Segment Routing enabled nodes. These nodes use the default SRGB label range, from 16000 to 23999, for the Prefix Segments. Node4 and Node5 do not have LDP enabled.

The node on the border of the two domains, Node3, has both Segment Routing and LDP enabled. It also uses the default SRGB [16000-23999]. The dynamic label range, used for e.g. LDP labels, starts at 24000, after the SRGB.

This section looks at the interworking from LDP to SR, for packets going from the LDP-only part of the network (blue) to the SR-only part of the network (green).

Node5 advertises its loopback prefix 1.1.1.5/32 with a Prefix-SID 16005. Node3 and Node4 program the MPLS forwarding entry for the Prefix Segment to Node5, the Prefix-SID label 16005.

Node1, Node2 and Node3 program their LDP forwarding entries for the FEC of destination Node5. LDP uses a downstream label allocation mode: an upstream node uses the LDP labels that are allocated and advertised by its downstream neighbor. LDP on Node3 dynamically allocates a local label 90007 for the FEC 1.1.1.5/32 and advertises this label binding to all its LDP neighbors. LDP on Node2 allocates local label 90100 – from the dynamic label range – for the FEC 1.1.1.5/32 and advertises this label binding to all its LDP neighbors. Finally, LDP on Node1 allocates a local label 90008 for the FEC 1.1.1.5/32.

Since Node4 is not LDP enabled, Node3 did not receive a LDP label binding from Node4 for prefix 1.1.1.5/32. Thus Node3 does not have an LDP outgoing label for the FEC 1.1.1.5/32. The LDP outgoing label is “Unlabelled”. However, Node3 has another Label Switched Path to Node5: the Prefix Segment of Node5. Node3 automatically connects the LDP Label Switched Path of the FEC 1.1.1.5/32 to the Prefix Segment of 1.1.1.5/32, providing a seamless Label Switched Path all the way from Node1 to Node5. No configuration is required for this functionality. Any node on the LDP/SR border automatically fulfills this interworking role.

7.2.1.1 LDP to SR Implementation Aspects

To understand how the Segment Routing and LDP entries end up in the forwarding table, a highly simplified diagram of interactions between the different components involved is used. This diagram is based on the Cisco IOS-XR implementation. This is the same diagram as used in section 7.1.3, “Implementation Aspects”.

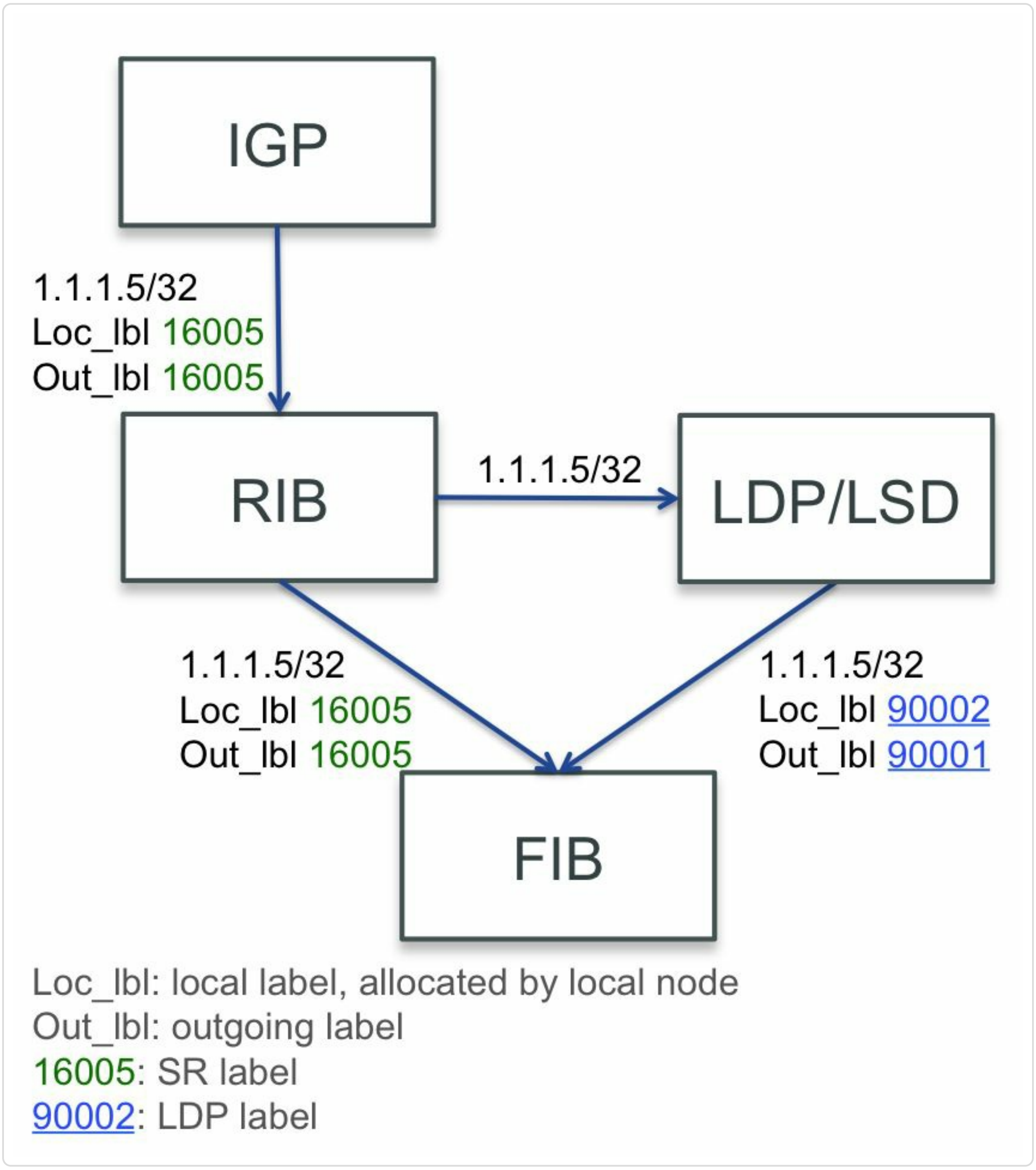


Figure 7-9: SR/LDP interworking – IGP and LDP programming FIB

First a quick reminder of the forwarding plane programming mechanism as described in section 7.1.3, "Implementation Aspects". When IGP installs its routes in RIB, it provides the prefix and paths with the Segment Routing labels to RIB. RIB sends the prefix path information to LDP/LSD and to FIB. LDP/LSD provides the LDP labels and sends the prefix path information to FIB. FIB receives both sources of information for the prefix path, from RIB and from LSD, and by default prefers the LDP/LSD provided information. The operator can configure the node to make FIB prefer the IGP/RIB provided information instead. For the prefix path received from the preferred source, the node installs both label imposition and label swap entries. For the prefix path received from the other, non-preferred source, the node only installs the label swap entry.

This is the general flow as was described before. But what happens to this flow if the downstream neighbor towards the destination prefix 1.1.1.5/32 is not LDP capable? In that case, LDP is not able to provide an outgoing label for the FEC bound to the prefix 1.1.1.5/32. Or what if the downstream neighbor is not Segment Routing capable? In that case, there is no outgoing label for the Prefix Segment.

In both the above cases, FIB does a "*replace*" operation using the two sources of path information, RIB and LDP/LSD. FIB replaces any unlabeled entry in the path information with the valid label provided by the other source of path information. The "replace" operation is also called "merge", but that is a misnomer. Labels are not merged, but replaced, as is shown further.

When FIB receives the prefix path information from the two sources, RIB and LSD, it replaces the "Unlabelled"[1] outgoing labels present in the prefix path entry of one source, with the valid outgoing label value of the corresponding prefix path from the other source. If FIB receives a prefix path entry from the LDP/LSD source with an "Unlabelled" outgoing label, as illustrated in Figure 7-10, then FIB replaces that label with the valid outgoing label value of the same prefix path from the IGP/RIB source. And Vice versa.

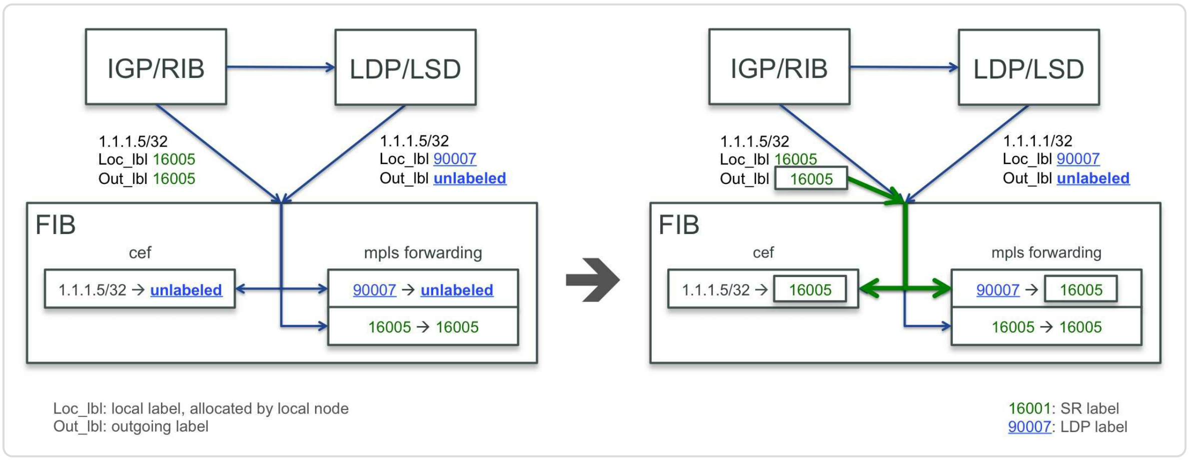


Figure 7-10: FIB "replace" operation – LDP to SR

This translates to the behavior that if no LDP outgoing label is available, the IGP/RIB Segment Routing provided outgoing label is used instead. This will happen if the downstream neighbor to the destination is not LDP enabled or if that downstream neighbor does not send a LDP label binding for the FEC bound to the destination. Similarly if the downstream neighbor to the destination is not Segment Routing enabled, then the LDP outgoing label for the FEC bound to the destination is used instead.

7.2.1.2 LDP to SR Illustration

Figure 7-11 repeats the topology in Figure 7-8, with an LDP-only part of the network on the left side (blue) and a Segment Routing only part of the network on the right side (green). Consider the path going from left to right, the path from LDP-only to Segment Routing-only. The programming of the forwarding table on Node3, the node on the border between the two network parts, is examined.

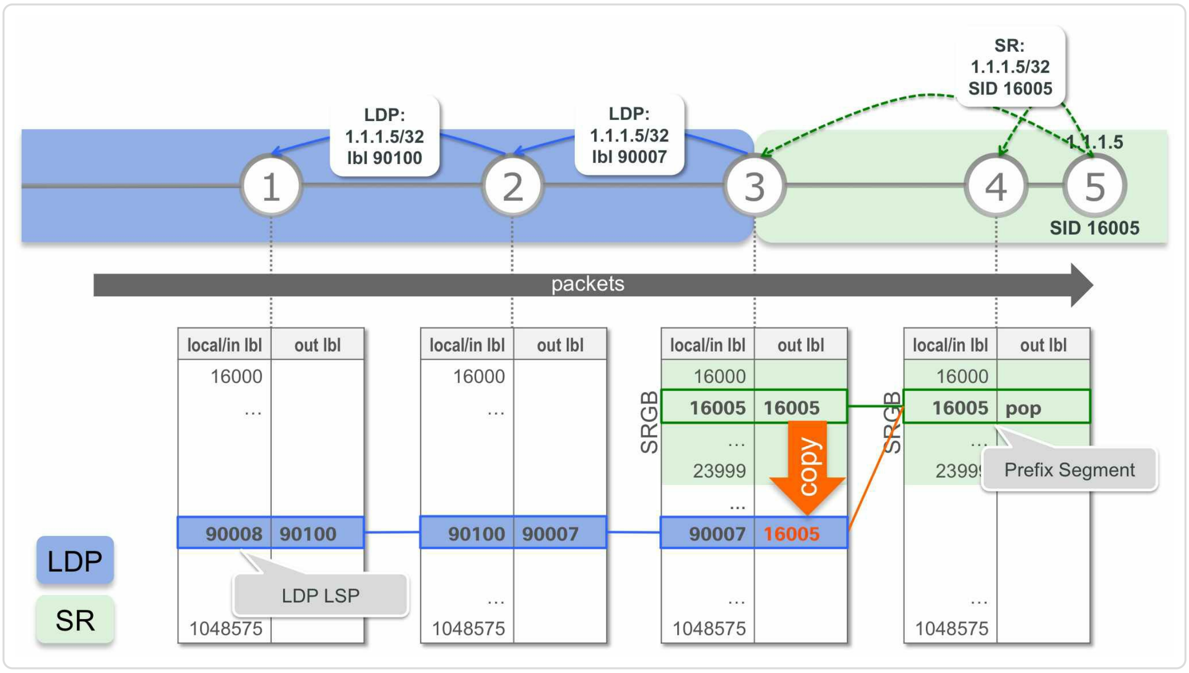


Figure 7-11: LDP to SR interworking illustration

Node4, the downstream neighbor of Node3 towards Node5, is not LDP enabled, which implies that Node3 has no outgoing LDP label for the FEC bound to destination Node5. FIB on Node3 automatically replaces the "Unlabelled" outgoing label from the LSD provided prefix path entry, by the valid outgoing label from the prefix path entry received from RIB. With this replace operation of the outgoing label, the LDP Label Switched Path is automatically stitched to the Prefix Segment path.

This behavior is now verified by examining the output of router show commands.

ISIS on Node3 has segment-routing mpls enabled without the `sr-prefer` keyword. This means that it uses the default label imposition preference: prefer LDP label imposition. See the configuration extract of Node3 in Example 7-11

Example 7-11: Node3 ISIS configuration extract

```
router isis 1
 address-family ipv4 unicast
  metric-style wide
  segment-routing mpls
 !
!
```

The output on Node3 of `show route detail` for the prefix 1.1.1.5/32 shows the RIB entry of the prefix. See the output in Example 7-12. This RIB prefix entry is installed by ISIS instance 1 (line 4). The prefix has a local label 16005 (line 16), which is the prefix-SID label 16005. There is only one path leading to this destination prefix, via interface GigabitEthernet0/0/0/0 and next-hop 99.3.4.4 (line 7). The outgoing label for this path is again the prefix-SID label 16005 (line 9). Both local and outgoing labels are Segment Routing labels, provided by IGP, which is ISIS in this example.

Example 7-12: Show route output on Node3

```
 1| RP/0/0/CPU0:xrvr-3#show route 1.1.1.5/32 detail
 2|
 3| Routing entry for 1.1.1.5/32
 4|   Known via "isis 1", distance 115, metric 40, labeled SR,
type level-2
 5|   Installed Feb 22 20:23:47.068 for 00:00:10
 6|   Routing Descriptor Blocks
 7|     99.3.4.4, from 1.1.1.5, via GigabitEthernet0/0/0/0
 8|       Route metric is 40
 9|       Label: 0x3e85 (16005)
10|       Tunnel ID: None
11|       Binding Label: None
12|       Extended communities count: 0
13|       Path id:1       Path ref count:0
14|       NHID:0x2(Ref:15)
15|   Route version is 0x224 (548)
16|   Local Label: 0x3e85 (16005)
17|   IP Precedence: Not Set
18|   QoS Group ID: Not Set
19|   Flow-tag: Not Set
```

```
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD
Type RIB_SVD_TYPE_LOCAL
  Download Priority 1, Download Version 5032
  No advertising protos.
```

The output on Node3 of `show mpls ldp bindings` for the FEC 1.1.1.5/32 shows the allocated local LDP label 90007. See the output in Example 7-13. Node4 is the upstream neighbor for Node3 towards destination 1.1.1.5/32 and Node3 should use the LDP label received from Node4 for FEC 1.1.1.5/32. Since LDP is not enabled on Node4, Node3 does not receive any label binding for FEC 1.1.1.5/32 from Node4. The output only shows a remote label binding for 1.1.1.5/32 from downstream neighbor 1.1.1.2 (Node2). Lacking a LDP label binding from Node4, Node3 has no LDP outgoing label to program for FEC 1.1.1.5/32.

Example 7-13: Show mpls ldp bindings output on Node3

```
RP/0/0/CPU0:xrvr-3#show mpls ldp bindings 1.1.1.5/32
1.1.1.5/32, rev 37
        Local binding: label: 90007
        Remote bindings: (1 peers)
            Peer                Label
            -----------------   ---------
            1.1.1.2:0           90100
```

The output of "`show mpls ldp forwarding`" for prefix 1.1.1.5/32 in Example 7-14 indeed shows an outgoing label "Unlabelled".

Example 7-14: Show mpls ldp forwarding output on Node3

```
RP/0/0/CPU0:xrvr-3#show mpls ldp forwarding 1.1.1.5/32

Codes:
  - = GR label recovering, (!) = LFA FRR pure backup path
  {} = Label stack with multi-line output for a routing path
  G = GR, S = Stale, R = Remote LFA FRR backup
```

```
Prefix           Label   Label(s)       Outgoing     Next
Hop             Flags
                 In      Out
Interface                        G S R
--------------- ------- -------------- ------------ -----------
-------- -----
1.1.1.5/32       90007   Unlabelled     Gi0/0/0/0
99.3.4.4
```

The default label imposition preference is used on Node3, which is to prefer the imposition of the LDP label if it is available. In the output of `show cef` for the prefix 1.1.1.5/32 in Example 7-15, one path is shown via outgoing interface Gi0/0/0/0 and next-hop 99.3.4.4 (line 7). The local label is the local label 90007 allocated for LDP FEC 1.1.1.5/32. The outgoing label is the Prefix-SID label 16005. This outgoing label is the result of the "replace" operation. The "unlabeled" outgoing LDP label for FEC 1.1.1.5/32 has been replaced by the valid Segment Routing outgoing label for 1.1.1.5/32: Prefix-SID label 16005.

Example 7-15: Show cef output on Node3

```
 1| RP/0/0/CPU0:xrvr-3#show cef 1.1.1.5/32
 2| 1.1.1.5/32, version 76, internal 0x1000001 0x5 (ptr
0xa13fa774) [1], 0x0 (0xa13dfc8c), 0xa28 (0xa1541898)
 3|  Updated Dec 18 12:35:42.562
 4|  local adjacency 99.3.4.4
 5|  Prefix Len 32, traffic index 0, precedence n/a, priority 15
 6|    via 99.3.4.4/32, GigabitEthernet0/0/0/0, 11 dependencies,
weight 0, class 0 [flags 0x0]
 7|     path-idx 0 NHID 0x0 [0xa10b13a0 0x0]
 8|     next hop 99.3.4.4/32
 9|     local adjacency
10|      local label 90007      labels imposed {16005}
```

The output of `show cef flags` for the prefix 1.1.1.5/32 in Example 7-16 shows that the `LDP/SR merge requested` flag is set (line 5), which means the merge or replace operation is requested for that prefix. The `LDP/SR merge active` flag is also set, which means the merge or replace operation actually happened.

Example 7-16: Show cef flags output on Node3

```
RP/0/0/CPU0:xrvr-3#show cef 1.1.1.5/32 flags
1.1.1.5/32, version 76, internal 0x1000001 0x5 (ptr
0xa13fa774) [1], 0x0 (0xa13dfc8c), 0xa28 (0xa1541898)
 leaf flags: owner locked, inserted

 leaf flags2: LDP/SR merge requested,LDP/SR merge active,
 leaf ext flags: PriChange,illegal-0x00000020,illegal-
0x00000200,illegal-0x00000800,
 Updated Dec 18 12:35:42.562
 local adjacency 99.3.4.4
 Prefix Len 32, traffic index 0, precedence n/a, priority 15
   via 99.3.4.4/32, GigabitEthernet0/0/0/0, 11 dependencies,
weight 0, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0xa10b13a0 0x0]
    next hop 99.3.4.4/32
    local adjacency
     local label 90007      labels imposed {16005}
```

An interesting detail: the `show cef detail` output shows the source of the cef entry. Example 7-17 shows the line in the output that contains the source. In this case the source is LSD (`source lsd`). `(5)` is the internal identifier for “LSD”.

Example 7-17: Show cef detail output on Node3

```
RP/0/0/CPU0:xrvr-3#show cef 1.1.1.5/32 detail | incl "source"
  gateway array (0xa14102b8) reference count 3, flags 0x68,
source lsd (5), 2 backups
```

The output of `show mpls forwarding` for label 16005 in Example 7-18 shows the MPLS forwarding table entry for the prefix-SID label associated with destination prefix 1.1.1.5/32. The local or incoming prefix-SID label 16005 is swapped with outgoing label 16005 and forwarded on outgoing interface Gi0/0/0/0, next-hop 99.3.4.4. No replace operation was done for this entry since the outgoing Prefix-SID label 16005 was readily available on Node3.

Example 7-18: show mpls forwarding output on Node3, SR entry

```
RP/0/0/CPU0:xrvr-3#show mpls forwarding labels 16005
Local  Outgoing    Prefix             Outgoing     Next
Hop        Bytes
Label  Label       or ID
  Interface                   Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16005  16005       SR Pfx (idx 5)     Gi0/0/0/0
99.3.4.4        0
```

The output of `show mpls forwarding` for the prefix 1.1.1.5/32, shows the LDP label 90007 as local label and the Prefix-SID label 16005 as outgoing label. This outgoing label is the result of the label replace (or merge) operation. Node3 replaces the original “Unlabelled” LDP outgoing label by the Prefix-SID label 16005. In the resulting forwarding entry the local LDP label 90007 is swapped with outgoing Prefix-SID label 16005 and forwarded on outgoing interface Gi0/0/0/0 with next-hop 99.3.4.4.

Example 7-19: show mpls forwarding output on Node3, LDP entry

```
RP/0/0/CPU0:xrvr-3#show mpls forwarding prefix 1.1.1.5/32
Local  Outgoing    Prefix             Outgoing     Next
Hop        Bytes
Label  Label       or ID
Interface                   Switched
------ ----------- ------------------ ------------ ------------
--- ------------
90007  16005       1.1.1.5/32         Gi0/0/0/0
99.3.4.4        0
```

Running a traceroute from Node1 to Node5 shows the LDP LSP being mapped to the Prefix Segment. See the output in Example 7-20. The dynamic LDP labels are used up to Node3 and there the label is swapped with the Prefix-SID label 16005. Remember that a classic traceroute displays the label stack of the packet as is received by the node replying to the probe message.

Example 7-20: traceroute on Node1 to Node5

```
RP/0/0/CPU0:xrvr-1#traceroute 1.1.1.5

Type escape sequence to abort.
Tracing the route to 1.1.1.5

 1  99.1.2.2 [MPLS: Label 90100 Exp 0] 19 msec  9 msec  9 msec
 2  99.2.3.3 [MPLS: Label 90007 Exp 0] 0 msec  0 msec  9 msec
 3  99.3.4.4 [MPLS: Label 16005 Exp 0] 9 msec  0 msec  0 msec
 4  99.4.5.5 0 msec  0 msec  9 msec
```

In the same topology, Node3 is now configured to prefer Segment Routing label imposition by adding the `sr-prefer` keyword to the segment-routing MPLS configuration of Node3. See the resulting configuration extract in Example 7-21.

Example 7-21: SR-prefer configuration

```
router isis 1
 address-family ipv4 unicast
  metric-style wide
  segment-routing mpls sr-prefer
 !
!
```

Now that Node3 is configured to prefer the Segment routing label imposition, the `show cef flags` output on Node3 for prefix 1.1.1.5/32 is shown in Example 7-22.

Example 7-22: Show cef flags output on Node3, no merge

```
 1 RP/0/0/CPU0:xrvr-3#show cef 1.1.1.5/32 flags
 2 1.1.1.5/32, version 5077, internal 0x1000001 0x83 (ptr
0xa13fa774) [1], 0x0 (0xa13dfe84), 0xa28 (0xa1541780)
 3  leaf flags: owner locked, inserted
 4
 5  leaf flags2: LDP/SR merge requested,RIB pref over LSD,SR
Prefix,
 6  leaf ext flags: PriChange,illegal-0x00000020,illegal-
0x00000200,illegal-0x00000800,
 7  Updated Dec 18 12:35:42.562
 8  local adjacency 99.3.4.4
 9  Prefix Len 32, traffic index 0, precedence n/a, priority 1
10    via 99.3.4.4/32, GigabitEthernet0/0/0/0, 9 dependencies,
weight 0, class 0 [flags 0x0]
11     path-idx 0 NHID 0x0 [0xa10b13a0 0x0]
12     next hop 99.3.4.4/32
13     local adjacency
14      local label 16005      labels imposed {16005}
```

The `LDP/SR merge requested` flag is set in the FIB entry (line 5), thus FIB will try to replace or merge the “Unlabelled” outgoing labels. But the `LDP/SR merge active` flag is *not* set; unset flags are not shown in the output. The replace operation did not actually take place here since there was no need. With the `sr-prefer` configuration, FIB prefers the Segment Routing label imposition and a valid Segment Routing outgoing label was already available. So, a replace operation was not needed.

Another flag is set in the cef entry as well: `RIB pref over LSD`. This flag indicates that Segment Routing label imposition (“RIB”) is preferred over LDP label imposition (“LSD”), which is indeed what was intended with the `sr-prefer` configuration.

The output of `show cef detail` on Node3 in Example 7-23 shows the source of the cef entry. Only the line in the output that contains the source is displayed. In this case, with `sr-prefer` configured, the source is RIB ("source rib").

Example 7-23: Show cef detail output on Node3, source RIB

```
RP/0/0/CPU0:xrvr-3#show cef 1.1.1.5/32 detail | i "source"
   gateway array (0xa141061c) reference count 3, flags 0x68,
source rib (7), 1 backups
```

The configuration of the label imposition preference on Node3 has no effect on then Label Switched Path from Node1 to Node5. A traceroute from Node1 to Node5 would for example show the same label values, regardless of the label imposition preference on Node3.

7.2.2 SR to LDP Interworking

Figure 7-12 shows the mirrored representation of the topology in Figure 7-7. It is the same topology, only the order of the nodes is reversed. The topology consists of a Segment Routing only part, on the left side, and an LDP only part on the right side. Node3, the node on the border between the two parts, runs both Segment Routing and LDP. The path from Segment Routing to LDP, from left to right, is examined. The Segment Routing nodes allocated SRGB [16000-23999].

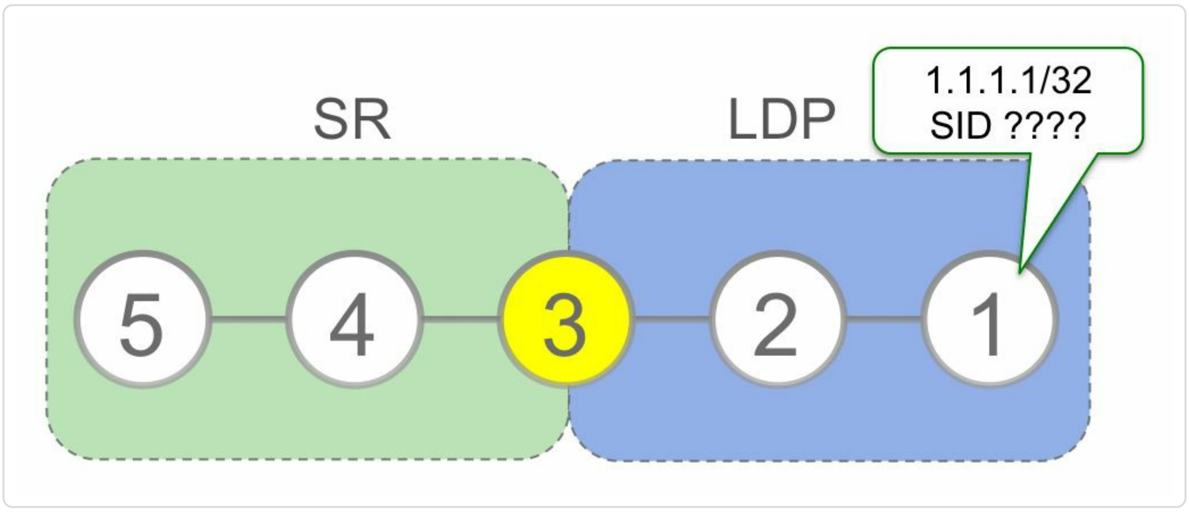


Figure 7-12: SR/LDP interworking network topology

Node5 is a Segment Routing node; it is located within the green Segment Routing area. If Node5 wants to use Segment Routing to transport packets to destination Node1, it must have a Prefix-SID label towards destination Node1. Since Node1 is not a Segment Routing node, it cannot advertise a Prefix-SID for its loopback prefix 1.1.1.1/32. Here the Mapping Server of chapter 8 comes into play. A Mapping Server is an entity (component, application or network node) that is capable of advertising a Prefix-SID associated with a prefix (a prefix-to-SID mapping) on behalf of other nodes. The Mapping Server functionality is built into Cisco IOS XR nodes.

A Mapping Server can advertise prefix-to-SID mappings in IGP on behalf of other nodes, including nodes that are not Segment Routing enabled. Since IGP floods these mappings in the network, all the nodes in the network receive these Mapping Server advertisements. The Segment Routing enabled nodes in the network use the prefix-to-SID mappings to program Segment Routing forwarding entries in their forwarding table. A node uses the Mapping Server advertised prefix-to-SID mapping for a prefix if no “native” Prefix-SID is available for that prefix. “Native” means that the node that originates the prefix also originates the associated Prefix-SID.

In the example, Node5 needs a Prefix-SID for the loopback prefix of Node1, 1.1.1.1/32. Since Node1 does not support Segment Routing, it does not advertise a Prefix-SID for its loopback prefix itself. A Mapping Server advertises a Prefix-SID index 1 associated with prefix 1.1.1.1/32 on behalf of Node1. Since there is no “native” prefix-SID available for 1.1.1.1/32, the Segment Routing nodes install the label derived from the Mapping Server advertised Prefix-SID index instead. The label value for Prefix-SID index 1 and SRGB [16000-23999] is 16001 (=16000 + 1).

Node4 and Node5, the Segment Routing nodes in the topology, install the following Segment Routing MPLS forwarding entry for the loopback prefix of Node1:

- The local or incoming label is the Prefix-SID label associated to the loopback prefix of Node1, 16001 in the example. This information comes from the Mapping Server.
- The outgoing label is the Prefix-SID label associated to the loopback prefix of Node1, 16001 in the example. This information comes from the Mapping Server.
- The outgoing interface is the interface to the downstream neighbor on the shortest path towards Node1. This information comes from the regular link-state IGP advertisements.

Node4 and Node5, also install the following Segment Routing label imposition entry for the loopback prefix of Node1:

- Loopback prefix of Node1, 1.1.1.1/32 in the example.
- The outgoing label is the Prefix-SID label associated to the loopback prefix of Node1, 16001 in the example. This information comes from the Mapping Server.
- The outgoing interface is the interface to the downstream neighbor on the shortest path towards Node1.

A Mapping Server advertises for example a Prefix-SID index 1 for 1.1.1.1/32, on behalf of Node1. This enables the use of Segment Routing labels for carrying traffic towards this destination within the Segment Routing domain.

Next problem to solve is the interworking from Segment Routing to LDP.

Node3, the node on the border between the two areas, runs both Segment Routing and LDP. Node3 is Segment Routing enabled, but its downstream neighbor, Node2, towards the destination Node1, is not Segment Routing enabled. Therefore Node3 cannot program a Segment Routing outgoing label for destination Node1 since Node2 would not be able to correctly interpret the Segment Routing label on packets it would receive. Instead of programming an “Unlabelled” entry in the forwarding table for destination Node1, Node3 automatically connects the Prefix Segment of Node1 to the LDP Label Switched Path towards Node1. Any node on the Segment Routing to LDP border automatically installs such SR-to-LDP forwarding entries. No configuration is required.

To connect the Prefix Segment to the LDP LSP, Node3 installs the following Segment Routing to LDP forwarding entry:

- The local or incoming label is the local Prefix-SID label associated with Node1’s loopback prefix, 16001 in this example. This information comes from the Mapping Server.
- The outgoing label is the LDP outgoing label for the FEC bound to destination Node1, as received from the downstream neighbor Node2.
- The outgoing interface is the interface towards neighbor Node2, the downstream neighbor on the shortest path to Node1.

Figure 7-13 illustrates the MPLS forwarding table programming in a network with Segment Routing interworking to LDP. It is the same topology as in Figure 7-8, but the order of the nodes is reversed in the presentation. The figure shows the path from Node5, on the left, to destination Node1, on the right. The illustration shows the MPLS forwarding tables under the nodes. The available label range is 16000 to 1 million. The left column of the forwarding tables shows the local or incoming labels, the right column shows the outgoing labels.

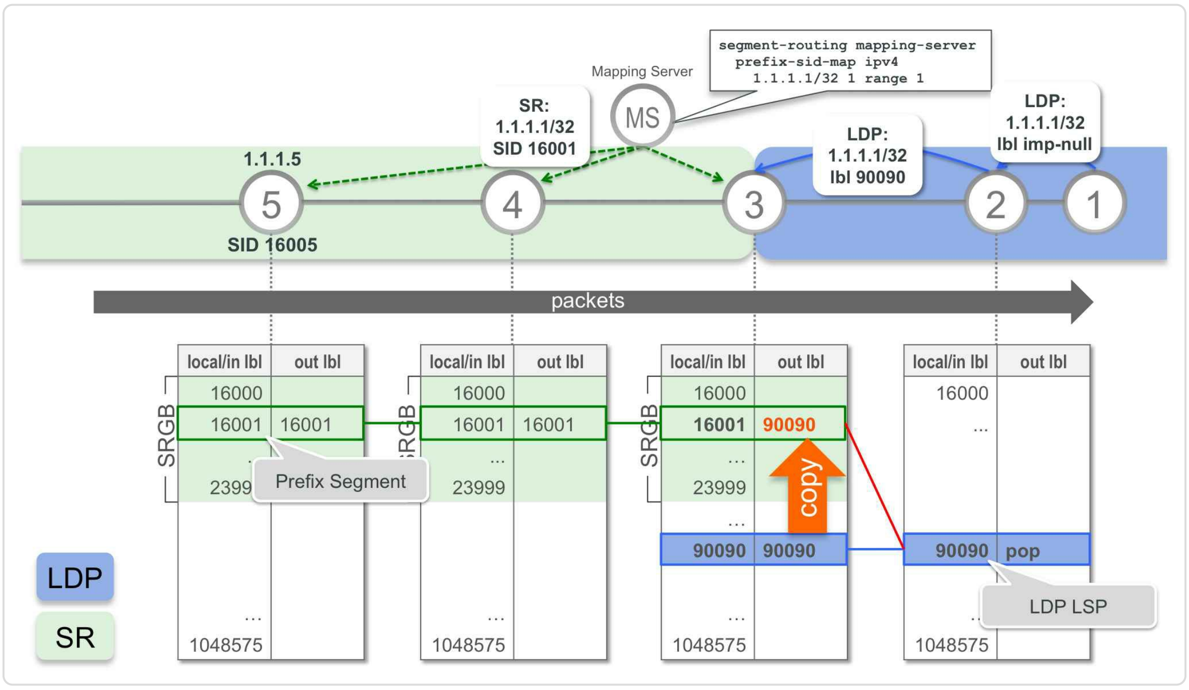


Figure 7-13: SR to LDP interworking illustration

Node4 and Node5 are Segment Routing enabled nodes. These nodes are using the default SRGB label range, from 16000 to 23999. Node4 and Node5 do not have LDP enabled.

Node1 and Node2 are legacy LDP-only nodes; they don't support Segment Routing. Thus Node1 and Node2 don't have an SRGB. It is assumed in this example that their dynamic label range starts at 24000.

The node on the border of the two domains, Node3, has both LDP and Segment Routing enabled. It also uses the default SRGB [16000-23999]. The dynamic label range, used for e.g. LDP labels, starts at 24000, after the SRGB.

Since its loopback prefix is locally attached, Node1 advertises the implicit-null LDP label binding for the FEC bound to its loopback prefix 1.1.1.1/32. Node2 allocates a local label 90090 – from the dynamic label range – for the FEC bound to prefix 1.1.1.1/32, and advertises this label binding to its LDP neighbors. Node2 installs the LDP forwarding entry: incoming label: 90090, outgoing label: pop, outgoing interface: Node1, the next-hop on shortest path to 1.1.1.1/32.

Node3 allocates a local label 90002 – from the dynamic label range – for the LDP FEC bound to prefix 1.1.1.1/32. Node3 installs the LDP forwarding entry: incoming label: 90002, outgoing label: 90090, outgoing interface: Node2, next-hop on shortest path to 1.1.1.1/32.

A Mapping Server advertises a prefix-SID index 1, associated to prefix 1.1.1.1/32. All Segment Routing enabled nodes use the Mapping Server's advertised prefix-SID index to program the Segment Routing label entries to 1.1.1.1/32. As all Segment Routing nodes in the topology use the default SRGB [16000-23999], they install the Prefix-SID forwarding entries using label 16001 (= 16000 + 1).

Since Node2 is not Segment Routing enabled, Node3 has no outgoing Prefix-SID label for 1.1.1.1/32.

However, Node3 has another label switched path towards 1.1.1.1/32: the LDP Label Switched Path for the FEC bound to 1.1.1.1/32. Node3 has received the outgoing label for that FEC from Node2: label 90090.

Node3 automatically connects the Prefix Segment of 1.1.1.1/32 to the LDP Label Switched Path for the FEC 1.1.1.1/32. This way, a seamless Label Switched Path is provided from Node5 all the way to Node1.

7.2.2.1 SR to LDP Implementation Aspects

Referring to the topology of Figure 7-13, the nodes in the Segment Routing-only part of the network need a Prefix-SID for the destination prefix in order to transport traffic to that destination using the Prefix Segment. Since the destination Node1 is not Segment Routing enabled, it cannot advertise a Prefix-SID itself. A Mapping Server is used to advertise a prefix-SID index 1 for the prefix 1.1.1.1/32 on behalf of Node1. The IGP on each Segment Routing node uses the prefix-to-SID mappings advertised by the Mapping Server to install the SR forwarding entries for the LDP-only nodes. All this is done in the control plane; for example the Prefix-SID label 16001 is installed in FIB the same way as a "native" Prefix-SID, via IGP/RIB.

The "replace" mechanism to install SR to LDP interworking forwarding entries is the same as for LDP to SR entries. Refer to section 7.2.1.1.

Figure 7-14 illustrates the behavior of Node3 in the topology of Figure 7-13. Node2 is the downstream neighbor of Node3 towards destination 1.1.1.1/32. Since Node2 is not Segment Routing enabled, IGP on Node3 does not have an outgoing Prefix-SID label for 1.1.1.1/32. Therefore IGP provides an "Unlabelled" outgoing label for the prefix 1.1.1.1/32 to RIB. Node2 allocates and advertises a LDP local label for the FEC 1.1.1.1/32: label 90090. LSD provides this label 90090 as outgoing label for the FEC 1.1.1.1/32 to FIB.

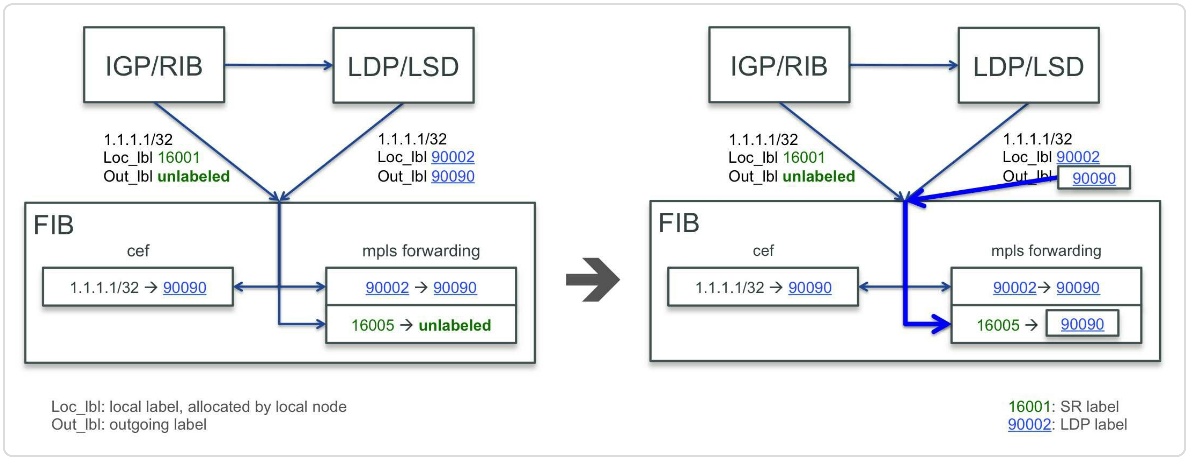


Figure 7-14: FIB "replace" operation – SR to LDP

FIB automatically replaces the "Unlabelled" outgoing label it received from RIB by the valid outgoing label 90090 it received from LDP/LSD. After this replace operation, the Prefix Segment to 1.1.1.1/32 is stitched to the LDP Label Switched Path of FEC 1.1.1.1/32.

7.2.2.2 SR to LDP Illustration

Node3, Node4, and Node5 install the Segment Routing forwarding entries to the LDP-only nodes as advertised by the Mapping Server. The Mapping Server advertises that Prefix-SID index 1 is mapped to prefix 1.1.1.1/32. See the output in Example 7-24. The "Range" column indicates the number of prefixes that are in this prefix-to-SID mapping entry; one in this example. See the Mapping Server section for more details.

Example 7-24: Mapping Server advertised prefix-to-SID mapping

```
RP/0/0/CPU0:iosxrv-5#show isis segment-routing prefix-sid-map
active

IS-IS 1 active policy
Prefix                SID Index    Range        Flags
1.1.1.1/32            1            1

Number of mapping entries: 1
```

ISIS on Node5 installs the prefix 1.1.1.1/32 in RIB, with the labels derived from the Mapping Server advertised Prefix-SID index. The calculation to derive the Prefix-SID label is same as for a “regular” Prefix-SID: add Prefix-SID index to the SRGB Base. In this example the Prefix-SID label is 16001 (= 16000 + 1). See output in Example 7-25. The route is marked as received from Mapping Server “SR(SRMS)” (line 4). The route has 16001 as local and outgoing labels (lines 9 and 16). Not illustrated here, but Node4 installs the equivalent entry in its routing table.

Example 7-25: Show route output on Node5 for SRMS advertised prefix-to-SID mapping

```
 1 RP/0/0/CPU0:iosxrv-5#show route 1.1.1.1/32 detail
 2
 3 Routing entry for 1.1.1.1/32
 4   Known via "isis 1", distance 115, metric 40, labeled
SR(SRMS), type level-2
 5   Installed Feb 23 11:16:58.647 for 00:21:50
 6   Routing Descriptor Blocks
 7     99.4.5.4, from 1.1.1.1, via GigabitEthernet0/0/0/1
 8       Route metric is 40
 9       Label: 0x3e81 (16001)
10       Tunnel ID: None
11       Binding Label: None
12       Extended communities count: 0
13       Path id:1       Path ref count:0
14       NHID:0x4(Ref:9)
15   Route version is 0x245 (581)
16   Local Label: 0x3e81 (16001)
17   IP Precedence: Not Set
18   QoS Group ID: Not Set
19   Flow-tag: Not Set
20   Fwd-class: Not Set
21   Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD
Type RIB_SVD_TYPE_LOCAL
22   Download Priority 1, Download Version 4622
23   No advertising protos.
```

So, the SR-only nodes can now forward SR traffic towards the LDP-only nodes.

Node3, on the border between SR-only and LDP-only parts of the network installs SR to LDP forwarding entries for the LDP-only prefixes. Example 7-26 shows the output of `show route detail` on Node3 for the prefix 1.1.1.1/32, the loopback prefix of Node1. This output shows the local and outgoing labels for each prefix path. In this example, there is only a single path to prefix 1.1.1.1/32: via Gi0/0/0/1 (line 7). The local label is 16001 (line 16), which is the Prefix-SID label for 1.1.1.1/32. IGP could not provide an outgoing label for prefix 1.1.1.1/32 since the downstream neighbor, Node2, is not Segment Routing enabled. The “Unlabelled” outgoing label is represented by `Label: None` in this output (line 9).

Example 7-26: Show route detail output on Node3

```
 1 RP/0/0/CPU0:xrvr-3#show route 1.1.1.1/32 detail
 2
 3 Routing entry for 1.1.1.1/32
 4   Known via "isis 1", distance 115, metric 20, labeled
SR(SRMS), type level-2
 5   Installed Feb 23 11:11:32.319 for 00:35:55
 6   Routing Descriptor Blocks
 7     99.2.3.2, from 1.1.1.1, via GigabitEthernet0/0/0/1
 8       Route metric is 20
 9       Label: None
10       Tunnel ID: None
11       Binding Label: None
12       Extended communities count: 0
13       Path id:1       Path ref count:0
14       NHID:0x3(Ref:4)
15   Route version is 0x25d (605)
16   Local Label: 0x3e81 (16001)
17   IP Precedence: Not Set
18   QoS Group ID: Not Set
19   Flow-tag: Not Set
20   Fwd-class: Not Set
```

```
  Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD
Type RIB_SVD_TYPE_LOCAL
  Download Priority 1, Download Version 5600
  No advertising protos.
```

The output of `show mpls ldp bindings` for FEC 1.1.1.1/32 in Example 7-27 shows the local LDP label 90002 that Node3 allocated for FEC 1.1.1.1/32, and the outgoing label for that FEC, label 90090, received from the downstream neighbor 1.1.1.2 (Node2).

Example 7-27: Show mpls ldp bindings output on Node3

```
RP/0/0/CPU0:xrvr-3#show mpls ldp bindings 1.1.1.1/32
1.1.1.1/32, rev 40
        Local binding: label: 90002
        Remote bindings: (1 peers)
            Peer                Label
            -----------------   ---------
            1.1.1.2:0           90090
```

At first, the label imposition preference on Node3 is the default, which is to prefer the imposition of the LDP label for incoming unlabeled packets.

The output of `show cef flags` for prefix 1.1.1.1/32 in Example 7-28 shows that the `LDP/SR merge requested` flag is set (line 5) but the `LDP/SR merge active` flag is not set (unset flags are not shown in the output). This means that the merge or replace operation was requested but did not actually happen. There was no need to replace the outgoing label since LDP provided a valid outgoing label to FIB. The `labels imposed` field in output shows that the LDP outgoing label 90090 is imposed, as expected.

Example 7-28: Show cef flags output on Node3

```
1 RP/0/0/CPU0:xrvr-3#show cef 1.1.1.1/32 flags
```

```
1.1.1.1/32, version 371, internal 0x1000001 0x1 (ptr
0xa13fa4f4) [1], 0x0 (0xa13dfd1c), 0xa28 (0xa15410f0)
 leaf flags: owner locked, inserted

 leaf flags2: LDP/SR merge requested,
 leaf ext flags: PriChange,illegal-0x00000020,illegal-
0x00000200,illegal-0x00000800,
 Updated Feb 23 12:09:41.629
 local adjacency 99.2.3.2
 Prefix Len 32, traffic index 0, precedence n/a, priority 3
   via 99.2.3.2/32, GigabitEthernet0/0/0/1, 9 dependencies,
weight 0, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0xa10b17e4 0x0]
    next hop 99.2.3.2/32
    local adjacency
     local label 90002      labels imposed {90090}
```

Example 7-30 shows the `show mpls forwarding` output for prefix 1.1.1.1/32. The output shows that the incoming LDP local label 90002 is swapped with the outgoing LDP label 90090 and forwarded on interface Gi0/0/0/1, next-hop Node2. Nothing special here, since an outgoing LDP label was readily available for destination 1.1.1.1/32.

Example 7-29: Show mpls forwarding output on Node3, LDP entry

```
RP/0/0/CPU0:xrvr-3#show mpls forwarding prefix 1.1.1.1/32
Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
Label  Label       or ID
Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
90002  90090       1.1.1.1/32         Gi0/0/0/1
99.2.3.3        0
```

Now Node3 is configured to prefer the Segment Routing label imposition (by configuring `sr-prefer`). The `show cef flags` output now looks as shown in Example 7-30. The `LDP/SR merge requested` flag is set in the FIB entry (line 5), and the `LDP/SR merge active` flag is set as well, which means the label replace operation is requested and the label has actually been replaced. The local label is the Prefix-SID label 16001 and the `labels imposed` is the outgoing LDP label 90090 for FEC 1.1.1.1/32. This outgoing label is the result of the label replace operation in FIB which replaced the Segment Routing "Unlabelled" outgoing label by the valid LDP outgoing label.

Another flag is set in the FIB entry: `RIB pref over LSD`. This flag is set when FIB must prefer the Segment Routing label imposition ("RIB") over the LDP for label imposition ("LSD"). This is the case if `sr-prefer` is configured.

Example 7-30: Show cef flags output on Node3 with sr-prefer

```
 1| RP/0/0/CPU0:xrvr-3#show cef 1.1.1.1/32 flags
 2| 1.1.1.1/32, version 5600, internal 0x1000001 0x87 (ptr
0xa13fa4f4) [1], 0x0 (0xa13dfef0), 0xa28 (0xa1541870)
 3|  leaf flags: owner locked, inserted
 4|
 5|  leaf flags2: LDP/SR merge requested,RIB pref over
LSD,LDP/SR merge active,SR Prefix,
 6|  leaf ext flags: PriChange,illegal-0x00000020,illegal-
0x00000200,illegal-0x00000800,
 7|  Updated Dec 18 12:35:42.563
 8|  local adjacency 99.2.3.2
 9|  Prefix Len 32, traffic index 0, precedence n/a, priority 15
10|    via 99.2.3.2/32, GigabitEthernet0/0/0/1, 9 dependencies,
weight 0, class 0 [flags 0x0]
11|     path-idx 0 NHID 0x0 [0xa10b17e4 0x0]
12|     next hop 99.2.3.2/32
13|     local adjacency
14|      local label 16001      labels imposed {90090}
```

The MPLS forwarding entry for incoming label 16001, the Prefix-SID label of 1.1.1.1/32, is shown in Example 7-31. As a result of the replace operation, the outgoing label for this entry is the LDP outgoing label for FEC 1.1.1.1/32.

Example 7-31: Show mpls forwarding output on Node3, SR entry

```
RP/0/0/CPU0:xrvr-3#show mpls forwarding labels 16001
Local  Outgoing    Prefix             Outgoing     Next
Hop        Bytes
Label  Label       or ID
Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
16001  90090       SR Pfx (idx 1)     Gi0/0/0/1
99.2.3.2        0
```

Note that the label swap entries, the label cross-connects for both Segment Routing and LDP local labels are always installed, regardless of the `sr-prefer` configuration.

A traceroute from Node5 to Node1 in Example 7-32 shows the shift from the Prefix Segment to the LDP LSP on Node3. Initially the packet has label 16001, which is the Prefix-SID label for 1.1.1.1/32. Then after Node3 the packet has an LDP label.

Example 7-32: Traceroute from Node5 to Node1

```
RP/0/0/CPU0:iosxrv-5#traceroute 1.1.1.1

Type escape sequence to abort.
Tracing the route to 1.1.1.1

 1  99.4.5.4 [MPLS: Label 16001 Exp 0] 29 msec  9 msec  9 msec
 2  99.3.4.3 [MPLS: Label 16001 Exp 0] 9 msec  9 msec  9 msec
 3  99.2.3.2 [MPLS: Label 90090 Exp 0] 9 msec  9 msec  9 msec
 4  99.1.2.1 9 msec  9 msec  9 msec
```

7.2.3 SR over LDP, LDP over SR

To interconnect two Segment Routing parts of the network across an LDP only part of the network, SR/LDP interworking is used on both border nodes connecting the network parts, as illustrated in Figure 7-15 (a). The Prefix Segment is connected to the LDP Label Switched Path at the SR-to-LDP boundary. And at the LDP-to-SR boundary, the LDP Label Switched Path is connected to the Prefix Segment.

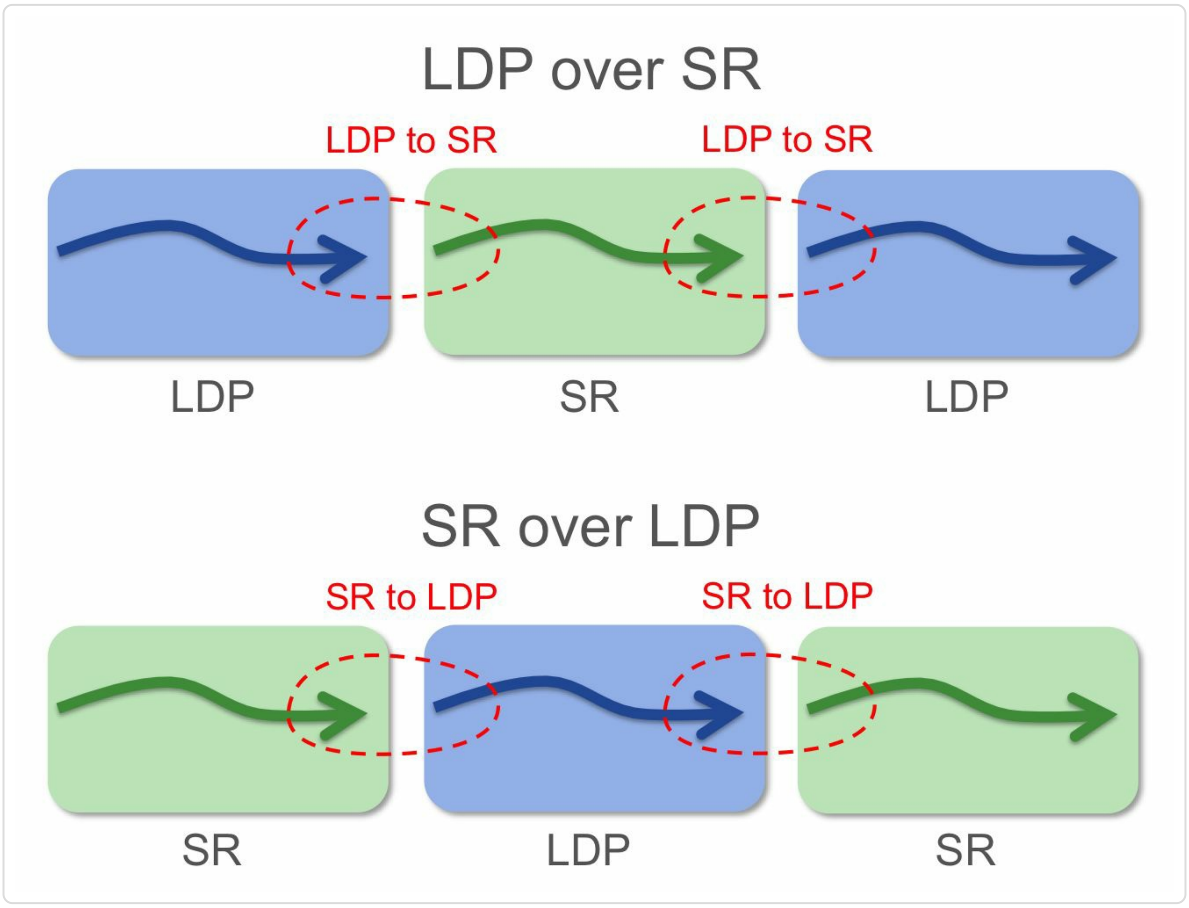


Figure 7-15: SR over LDP and LDP over SR

Is a Mapping Server needed to interconnect two Segment Routing parts of the network over an LDP-only part of the network?

If no Segment Routing transport is required from sources in the Segment Routing part of the network to destinations that are located in the LDP-only part of the network, then a Mapping Server is not strictly required.

The Segment Routing source node knows the prefix-SID of the destination node. The destination node is Segment Routing enabled and natively advertises a Prefix-SID for its loopback prefix. See Figure 7-16.

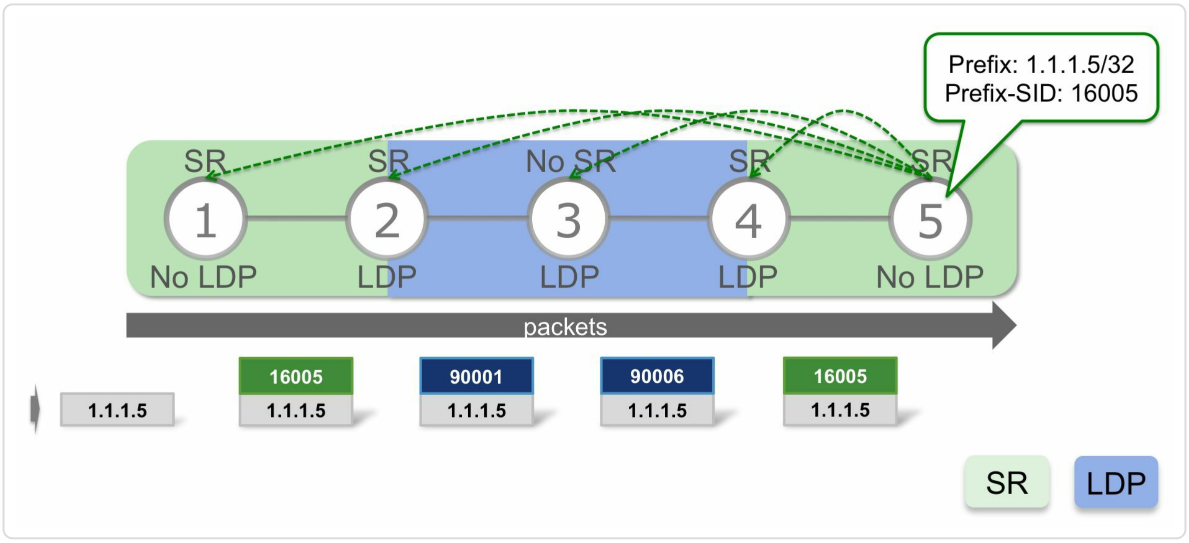


Figure 7-16: SR over LDP

If Segment Routing transport is required from sources in the Segment Routing part of the network to destinations that are located in the LDP-only part of the network, then a Mapping Server is needed. This is the regular SR to LDP interworking which requires a Mapping Server.

To interconnect two LDP-only parts of the network across a Segment Routing only part of the network, the LDP Label Switched Path is connected to the Prefix Segment at the LDP-to-SR boundary. At the SR-to-LDP boundary, the Prefix Segment is connected to the LDP Label Switched Path. See Figure 7-15 (b).

Is a Mapping Server needed to interconnect two LDP-only parts of the network over a Segment Routing part of the network? The answer is: Yes, always. Inside the Segment Routing network part, prefix-SID labels are needed to transport the labeled packets. If a source in the LDP-only part of the network needs to transport packets to a destination in the LDP-only part of the network across the Segment Routing network, then a Mapping Server is required. The nodes in the Segment Routing network need a Prefix-SID label for the destination prefix, which is originated by an LDP-only node. This LDP-only node cannot advertise a Prefix-SID for its loopback prefix. A Mapping Server advertises a Prefix-SID for that loopback prefix on behalf of the LDP-only node. See illustration in Figure 7-17.

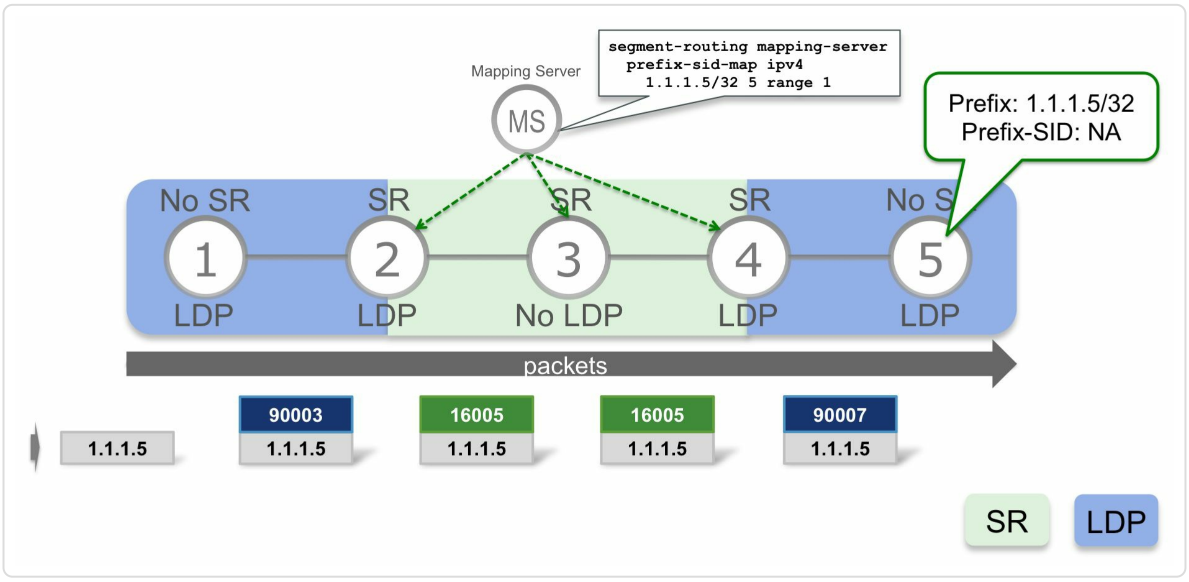


Figure 7-17: LDP over SR

If a source in the LDP-only part of the network needs to transport packets to a destination in the Segment Routing part of the network, then a Mapping Server is not strictly required. This is the regular LDP to SR interworking functionality.

The SR/LDP interworking functionality is also applied on the Topology Independent-LFA backup path, see chapter 9.

The SR/LDP interworking implementation tries to keep packets on the same transport type, be it Segment Routing or LDP, where possible. Only if needed, SR/LDP interworking functionality is used to interconnect Prefix Segments with LDP Label Switched Paths.

In Figure 7-18, a packet that is carried on an LDP LSP arrives on Node1. The packet traverses a Segment Routing-only island, represented by Node3. LDP is not enabled on Node3.

SR/LDP interworking functionality on Node2 connects the LDP LSP to the Prefix Segment to the destination. From there the packet stays on the Prefix Segment as long as there is continuous Segment Routing connectivity. Transport only switches back to an LDP LSP if required, if an LDP-only node is traversed. The same is true for the reverse case. A packet stays on an LDP LSP as long as there is continuous LDP connectivity.

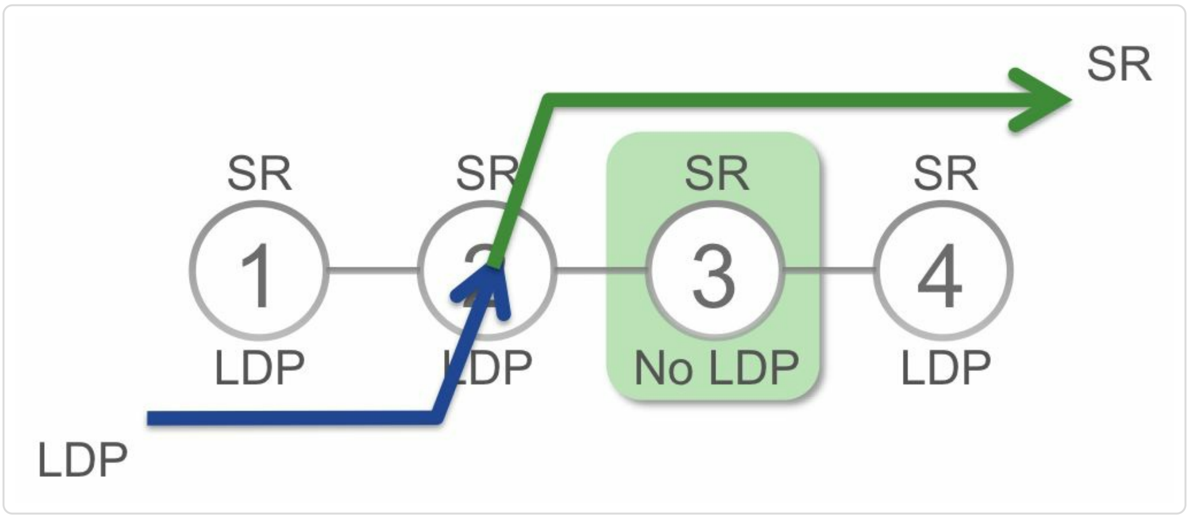


Figure 7-18: SR/LDP interworking only if required

HIGHLIGHT: Deployment Considerations

It is recommended to include the following prefixes belonging to non-Segment Routing capable (i.e. LDP) nodes in the Prefix-SID mappings of the Mapping Server:

- all loopback prefixes on these nodes which are used to terminate services
- all loopback prefixes that are associated with the "router-id" on those nodes which terminate services (i.e. ingress or egress edge routers in the transport network)
- any other prefixes to which traffic/services could be destined to, but which are not really associated with Prefix-SIDs (e.g. destinations behind an edge router or redistributed prefixes) but for which we need features like TI-LFA protection in the Segment Routing domain.

7.2.4 SR/LDP Interworking Summary

Figure 7-19 summarizes the SR/LDP interworking functionality. The illustration is divided in three columns. The node on top of each column indicates the location of the node in the network. Each node is on the border between an LDP + Segment Routing part of the network on the left side and the different possible Segment Routing and/or LDP combinations on the right side. The packets underneath each node show how the border node handles packets that arrive (1) without label, (2) with a LDP transport label, and (3) with a Segment Routing transport label. Depending on which transport is available on the right side of the border node, three models are distinguished: "ships-in-the-night", "to-LDP-only", and "to-SR-only".

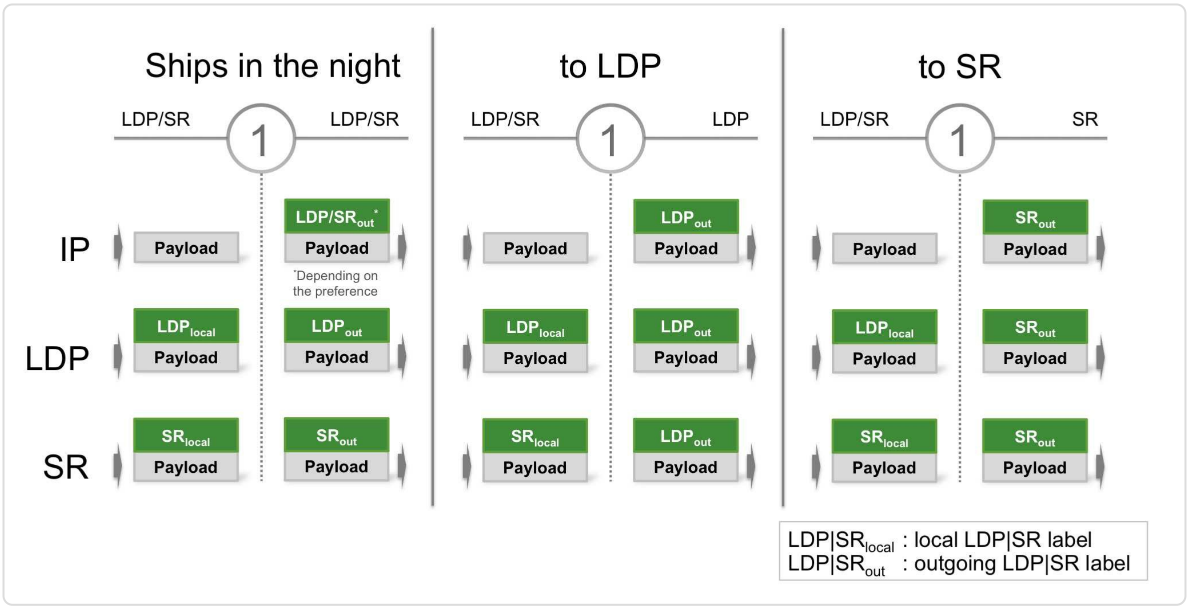


Figure 7-19: SR/LDP interworking summary

The first model is the “ships-in-the-night” model, shown in the first column. This model provides both Segment Routing and LDP connectivity on both sides of the border node. The second model is the “to-LDP-only” model, shown in the middle column, which has an LDP-only part of the network on the right side of the border node. The third model is the “to-SR-only” model, shown in the right column, which has a Segment Routing-only part of the network on the right side of the border node.

When an unlabeled IP packet arrives from the left (first row, named “IP”), the border node imposes the outgoing label for the destination prefix. For the ships-in-the-night model the imposed label depends on the label imposition preference configuration of the node since both Segment Routing and LDP transport labels are available. For the other models, only one outgoing label is available, which is then imposed on the outgoing packet.

When a packet with an LDP transport label arrives on the node (second row, named “LDP”), the label will be swapped with an LDP outgoing label for the “ship-in-the-night” and the “to-LDP-only” models. The LDP label is swapped with a Segment Routing label for the “to-SR-only” model.

When a packet with a Segment Routing transport label arrives on the node (third row, named “SR”), the label will be swapped with a Segment Routing outgoing label for the “ship-in-the-night” and the “to-SR-only” models. The Segment Routing label is swapped with a LDP label for the “to-LDP-only” model.

> “Segment Routing interworking and transition mechanisms have been key enablers for its deployment in existing MPLS networks. There is flexibility in using different models (ships-in-the-night and interworking) for different services as well as gradual upgrade of older routing platforms in the network.”
>
> *— Ketan Talaulikar*

7.3 Summary

- By default a Node imposes an LDP label on incoming unlabeled packets, Segment Routing imposition preference can be configured.
- A node installs both LDP and SR MPLS-to-MPLS forwarding entries (label swap) if available, irrespective of the SR/LDP preference configuration.
- The SR/LDP interworking data plane functionality is achieved by replacing an unknown outgoing label (Unlabelled) by a valid outgoing label from the other protocol.
- A Mapping Server advertises Prefix-SIDs on behalf of LDP-only nodes. This enables sending Segment Routing traffic to LDP-only nodes.
- A node on the border of an LDP part of the network to an SR-only part of the network automatically stitches LDP LSPs to Prefix Segments.
- A node on the border of an SR part of the network to an LDP-only part of the network automatically stitches Prefix Segments to LDP LSPs.

7.4 References

- [draft-ietf-spring-segment-routing-ldp-interop] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., and S. Litkowski, “Segment Routing interworking with LDP”, draft-ietf-spring-segment-routing-ldp-interop (work in progress), July 2016, https://datatracker.ietf.org/doc/draft-ietf-spring-segment-routing-ldp-interop.
- [draft-ietf-spring-segment-routing-mpls] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Shakir, R., Tantsura, J., and E. Crabbe, “Segment Routing with MPLS data plane”, draft-ietf-spring-segment-routing-mpls (work in progress), September 2016, https://datatracker.ietf.org/doc/draft-ietf-spring-segment-routing-mpls.

---

[1] British-style “Unlabelled” with double “l”, as is used in the show output of the router

8 SEGMENT ROUTING MAPPING SERVER

8.1 Function of Mapping Server

A Segment Routing Mapping Server, or simply “Mapping Server”, is an entity that is capable of advertising prefix to Prefix-SID index mappings in IGP. A Mapping Server can be a function on a network node, an independent application, or dedicated platform.

The Mapping Server functionality is defined for the SR/LDP interworking functionality.

The term “Segment Routing Mapping Server” is sometimes abbreviated to “SRMS”.

A non-Segment Routing enabled node cannot advertise a Prefix-SID associated with its prefix, simply because the node is not Segment Routing enabled. If a Prefix-SID is needed for that prefix, the advertisement of the Prefix-SID for that prefix must be done by another entity in the network, on behalf of the node that owns the prefix. The Mapping Server function provides that service, by advertising lists of prefixes with their associated Prefix-SID indexes in the IGP, the so-called “prefix to Prefix-SID index mappings”, or, somewhat shorter, “prefix-to-SID mappings”.

The Mapping Server functionality is required in the network to enable interworking between Segment Routing-enabled nodes and non-Segment Routing-enabled nodes, such as LDP-only nodes. Therefore the Mapping Server function is a key component of this SR/LDP interworking functionality. The SR/LDP interworking chapter 7, “Segment Routing in Existing MPLS Networks” explains how a Mapping Server fits in the SR/LDP interworking architecture. This chapter covers the Mapping Server functionality itself.

SRMS for advertising SIDs of SR-enabled node

“The SRMS concept has been invented for the SR/LDP interworking solution. It allows the SR nodes to discover the SID’s of LDP-only nodes (or prefixes from non-SR nodes).

Personally, I would not consider extending the SRMS to distribute the SIDs of SR-enabled nodes. SR-enabled nodes are capable of doing this and, in my opinion, should be doing it.

A key SR design principle is simplicity and hence we always seek to automate behaviors and simplify the operation.

One may ask himself whether configuring all the Prefix-to-SID mappings on some central nodes and distributing through the IGP could provide some further simplification. I am not convinced about this.

I do not see the configuration gain: in fact, the prefix-to-SID mapping needs to be configured twice or more with the SRMS option compared to the classic configuration on the SR node itself. This is even more obvious considering that the Prefix-to-SID mapping is expected to never change and hence it just needs to be configured once per SR node.

The SID of a prefix is the corner stone on which all of the solution is built. This corner stone must be as stable and reliable as possible and hence should not depend on any other node than the one advertising the prefix to which the SID is bound.

As networks evolve in the SDN era, automation and programmability are becoming key for their provisioning and management. Tools like Cisco Network Services Orchestrator (NSO)[1] facilitate a centralized and streamlined provisioning model which also makes tasks such as of IP address allocations (and as an extension their Prefix SIDs for SR) easier and in a manner that reduces errors.”

— Clarence Filsfils & Ketan Talaulikar

8.2 Position of Mapping Server in Network

A Mapping Server is a functional component of the network. Its purpose is to advertise prefix-to-SID mappings. The Mapping Server function can be realized as a module or component of an existing device (NMS, router, switch) or as a dedicated entity (or virtualized service) in the network. There may be, and likely will be, multiple Mapping Server entities in the network. This book only covers the Mapping Server as a component of a router (physical or virtual).

The position of the Mapping Server in the network is somewhat comparable to a BGP route-reflector.

Similar to a BGP route-reflector, the Mapping Server is a control plane functionality. A node running the Mapping Server functionality does not need to be in the data path, but it can be. The exact position of the Mapping Server in the network does not influence its function since the mapping advertisements are distributed using the regular IGP advertisement mechanism. Since it uses the IGP to distribute its mappings, a Mapping Server needs to have an IGP adjacency to the network. Nodes that do not support or understand the Mapping Server IGP advertisements will still forward these advertisements using the regular IGP flooding mechanism and we will look at this closer further in this section.

A Mapping Server does not need to be a node in the network that is dedicated to the Mapping Server function, but it can be. Any node that supports the Mapping Server functionality, such as any Cisco IOS XR router, can handle the Mapping Server function when configured to do so

The role of the Mapping Server in the network is crucial, so resiliency is needed and redundancy should be provided.

Positioning the SRMS function in the network

“There are certain practical aspects to position the SRMS function in the network, which could also serve as a guideline for its positioning.

This function is embedded in all routers with Cisco IOS-XR software to avoid need for additional or special platform and give flexibility in positioning.

This function is seen as part of the distributed control plane of Segment Routing since it involves flooding via IGPs. Embedding it in redundant routers in the network makes most sense.

For redundancy purposes, it needs to be placed on at least two routers which are themselves redundant (i.e. less likely to fail or get isolated together) and preferably platforms with good High Availability (HA) features. Two are normally sufficient in most networks.

Placing this function on a large number of devices could be counter-productive as one would need to ensure the consistency of the mappings across them and the possibility of errors increases especially when modifying mappings on large number of routers in a coordinated manner in a working production network.

Placing this function on devices like an OSPF ABR router will ensure that the mappings get propagated from a single source into the multiple areas concerned. These border routers are also generally redundant gateways between parts of the network through which a lot of services flow. Note that even if configured on other devices, the ABR would need to handle their propagation across areas. Note this does not apply to Cisco IOS XR ISIS implementation at the time of writing this book see section 8.10.1.”

— Ketan Talaulikar

Segment Routing Mapping Server is the function where the Prefix-SID mappings for prefixes belonging to non-SR enabled nodes are provisioned centrally and distributed via IGP across the network.

Typically enabled on two (or rarely more) redundant routers in the network.

8.3 Function of Mapping Client

As its name indicates, a Mapping Client is a client of a Mapping Server. The Mapping Client function is a function in the IGP to receive, parse and processes the prefix to Prefix-SID mappings advertised by the Mapping Servers. Nodes that need to install the Prefix Segment forwarding entries derived from the prefix-to-SID mappings, use the Mapping Client functionality.

A node that is a Mapping Server can at the same time be a Mapping Client. For example a Mapping Server that is on the data path. Since the Mapping Server function does not need to run on a dedicated node, it is possible to implement the Mapping Server function on a node in the network that is on the data path. Since it is on the data path, such a node would also use the prefix-to-SID mappings to install the forwarding entries derived from these mappings; therefor it would need to be a Mapping Client. If a Mapping Client is also a Mapping Server, then the Mapping Client considers the mappings advertised by the local Mapping Server the same as it considers mappings of a remote Mapping Server. The Mapping Client acts as if all Mapping Servers are remotely located.

A Mapping Client receives prefix-to-SID mappings from all Mapping Servers in the network. The prefix-to-SID mappings it receives may be invalid or conflicting, due to software errors, human errors, during reconfiguration, etc. Therefore the Mapping Client uses a set of rules to select a valid set of mappings from all received prefix-to-SID mappings. The selection rules must result in a consistent set of prefix-to-SID mappings throughout the network: each Mapping Client in the network must come up with the same set of valid mappings. This is necessary to get consistent forwarding entries based on these prefix-to-SID mappings in order to avoid forwarding loops and blackholes. This consistent valid set of prefix-to-SID mappings is called the Active Mapping Policy.

An Active Mapping Policy is derived per IGP instance on a node. The IGP instance uses this Active Mapping Policy to derive a Prefix-SID for some or all prefixes that do not have an associated Prefix-SID.

The prefix-to-SID mapping entries that were received but did not make it to the Active Mapping policy, possibly due to overlaps or conflicts with other prefix-to-SID mapping entries, are inserted into the Backup Mapping Policy.

If one or more Mapping Servers are deployed in the network, then all Segment Routing enabled nodes should use the prefix-to-SID mappings advertised by these Mapping Servers. If not all Segment Routing enabled nodes do so, then this may lead to forwarding entries that are not consistent between nodes that use the mappings and nodes that do not use the mappings. This may lead to traffic blackholes or forwarding loops. Therefore, a common sense design rule is: "If a Mapping Server is deployed in the network, all Segment Routing nodes in the network should be a Mapping Client." For this practical reason, the IGPs in Cisco IOS-XR implementations will by default operate as Mapping Clients and accept mappings coming from Mapping Servers – this can be changed via explicit configuration.

HIGHLIGHT: Segment Routing Mapping Client

Segment Routing Mapping Client is the function on Segment Routing enabled IGP nodes which accepts and uses the Prefix-SID mappings advertised by the Segment

Routing Mapping Server(s) for assigning SIDs to prefixes for which it does not have “native” IGP Prefix-SID advertisements.

Typically enabled by default on all Segment Routing capable IGP routers.

8.4 Mapping Server Architecture

Figure 8-1 shows a simplified diagram of the Cisco IOS XR Mapping Server architecture that can serve as a good reference for understanding the concept. On the left hand side you see the Mapping Manager component, on the right hand side the IGP component. The Mapping Manager validates the local prefix-to-SID mapping configuration and provides a valid Local Mapping Policy to the IGP. The different components in the architecture are examined by walking through its functionality.

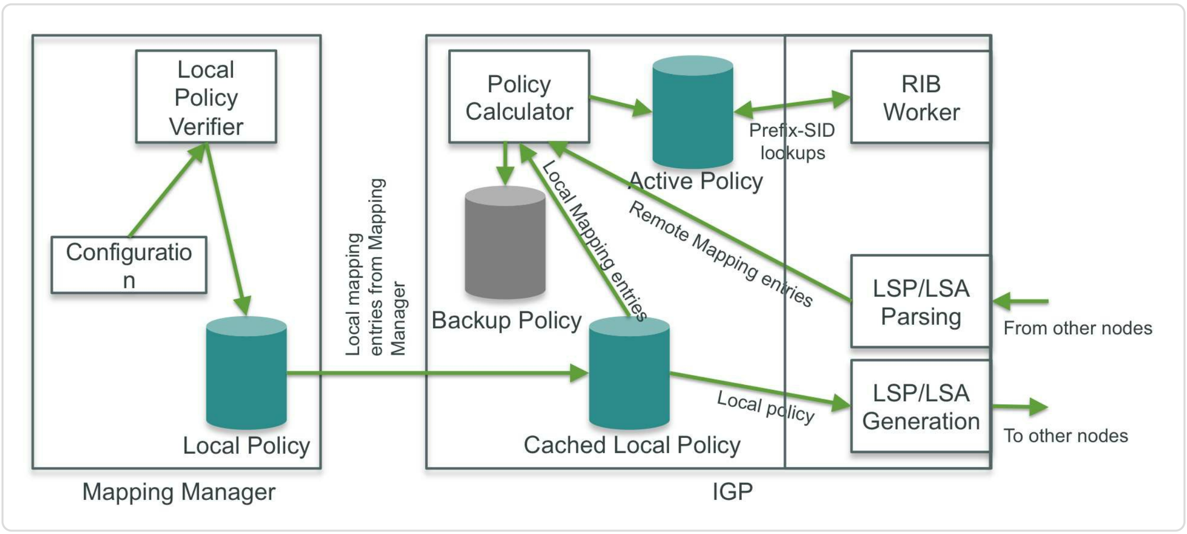


Figure 8-1: Mapping Server Architecture

The Mapping Manager functionality starts by configuring the prefix-to-SID mappings on the Mapping Server. The prefix-to-SID mappings are configured under the global segment-routing configuration in Cisco IOS XR.

The Local Policy Verifier verifies the configuration for consistency and correctness and then inserts the valid consistent prefix-to-SID mappings into a Local Mapping Policy database.

The IGP component caches the Local Mapping Policy and this Local Policy cache is used by the IGP Mapping Server functionality to generate the IGP updates to distribute the prefix-to-SID mappings throughout the network. The Local Policy cache is updated when the Mapping Server configuration is updated. Different IGP instances running on a node connect to the central Mapping Manager to retrieve the global Local Mapping Policy database.

The Mapping Client functionality also uses the Local Mapping Policy if the Mapping Client is at the same time a Mapping Server. Other Mapping Clients, those that are not running on a Mapping Server, only get prefix-to-SID mappings received from remote Mapping Servers.

Both prefix-to-SID mappings in the Local Policy and mappings received from remote Mapping Servers are fed into the Policy Calculator. The Policy Calculator validates the prefix-to-SID mappings and derives a valid and non-overlapping (see section 8.6, "Mapping Ranges Conflict Resolution") set of prefix-to-SID mappings that is consistent across all Mapping Clients in the network. This set of prefix-to-SID mappings is inserted into the Active Mapping Policy database.

The IGP instance uses this Active Mapping Policy database to derive a Prefix-SID for some or all prefixes that do not have an associated Prefix-SID. Note that each IGP instance running on the node derives its own, likely different Active Mapping Policy. While the Local Mapping Policy is global on the node, shared by all IGP instances.

8.5 Mapping Server Local Policy Configuratior

A Local Mapping Policy is statically configured on a node that is acting as Mapping Server. This Local Mapping Policy consists of static mappings of prefixes to prefix-SID indexes, the prefix-to-SID mappings. The Local Mapping Policy can contain both IPv4 and IPv6 prefix-to-SID mappings. The IPv6 prefix-to-SID mappings can only be advertised by ISIS in the Cisco IOS-XR implementation at the time of writing this book.

The prefix-to-SID mappings do not need to be configured per individual prefix, but they can be configured as ranges of prefixes mapping to ranges of prefix-SID indexes. Each prefix in the range then statically maps to the prefix-SID index at the same offset in the range.

The mapping configuration for a range of prefixes consists of the prefix, its prefix length, the first Prefix-SID index to be used for this prefix range, and the size of the prefix range. Note well that the configuration requires SID indexes, not label values. See Example 8-1.

Example 8-1: Mapping Server Local Policy configuration

```
segment-routing
 mapping-server
  prefix-sid-map
   address-family ipv4
    !! <prefix>/<mask> <first-SID-index> range <range>
    1.1.1.1/32 1 range 10
   !
   address-family ipv6
    2001::1:1:1:1/128 4001 range 10
   !
  !
 !
!
```

Each prefix in the range uses the same configured prefix length. The prefixes are typically host prefixes, /32 or /128, when we consider the SR/LDP interworking scenarios, but other prefix lengths are permitted.

Each line in this configuration results in a separate prefix-to-SID mapping advertised by IGP.

For ISIS, it is possible to indicate in the Mapping Server advertisements that a prefix is known to be connected or attached to the node that advertises it with the A-flag (Attached) set. See IGP control plane chapter 5, "Segment Routing IGP Control Plane". This functionality can be used if the Mapping Server advertises Prefix-SIDs on behalf of IGP nodes that are not Segment Routing nor LDP enabled. The "Attached" flag can be set in the Local Mapping Policy configuration.

Use the `show segment-routing mapping-server prefix-sid-map ipv4|ipv6 [detail]` command to verify the Local Mapping Policy. An individual prefix-to-SID mapping entry can be examined by adding the prefix to the above command.

Example 8-2 and Example 8-3 show the output of the Local Mapping Policy for the configuration of Example 8-1.

Example 8-2: IPv4 Local Mapping Policy

```
RP/0/0/CPU0:xrvr-4#show segment-routing mapping-server
prefix-sid-map ipv4
Prefix                    SID Index    Range         Flags
1.1.1.1/32                1            10

Number of mapping entries: 1

RP/0/0/CPU0:xrvr-4#show segment-routing mapping-server
prefix-sid-map ipv4 detail
```

```
Prefix
1.1.1.1/32
    SID Index:      1
    Range:          10
    Last Prefix:    1.1.1.10/32
    Last SID Index: 10
    Flags:

Number of mapping entries: 1
```

Example 8-3: IPv6 Local Mapping Policy

```
RP/0/0/CPU0:xrvr-4#show segment-routing mapping-server
prefix-sid-map ipv6
Prefix                                        SID Index
Range        Flags
2001::1:1:1:1/128                             4001
10

Number of mapping entries: 1

RP/0/0/CPU0:xrvr-4#show segment-routing mapping-server
prefix-sid-map ipv6 detail
Prefix
2001::1:1:1:1/128
    SID Index:      4001
    Range:          10
    Last Prefix:    2001::1:1:1:a/128
    Last SID Index: 4010
    Flags:

Number of mapping entries: 1
```

8.6 Mapping Ranges Conflict Resolution

Terminology:

- Mapping entry: prefix-to-SID mapping for an individual prefix
- Mapping range: prefix-to-SID mapping for a range of prefixes, as configured in the Local Policy. A mapping range can be expressed as a tuple (P/L, S, N), with P/L: first prefix and length in range; S: first SID index in range; and N: range size.

A Mapping Server advertises one or more mapping ranges and multiple Mapping Servers may exist in the network, each advertising its own mapping ranges. A Mapping Client considers all mapping ranges that it receives from all Mapping Servers to derive the Active Policy. The Active Policy consists of non-overlapping mapping ranges. This Active Policy should be the same on all Mapping Clients in the network to achieve consistent forwarding throughout the network for the Prefix-SIDs of the mapping entries.

Note that the Mapping Client considers mapping *ranges* in its selection process, not individual mapping *entries*.

Mapping ranges can overlap or conflict. Overlapping mapping ranges contain at least one mapping entry with a common prefix or SID index. Conflicting mapping ranges contain at least one entry that conflicts with another entry. The conflict can be a prefix conflict and/or a SID index conflict. If different SID indexes are assigned to the same prefix, then there is a "prefix conflict". If the same SID index is assigned to multiple prefixes, then there is a "SID index conflict".

Note that conflicting ranges are also overlapping since the conflicting ranges contain common prefixes and/or common SID indexes, otherwise they could not conflict. But overlapping ranges are not always conflicting. The Active Policy consists of non-overlapping, hence non-conflicting mapping ranges.

The Mapping Server configuration on Cisco IOS-XR only accepts non-overlapping mapping ranges for the Local Mapping Policy. Therefore, a Cisco IOS XR Mapping Server only advertises valid, non-overlapping mapping ranges. However, it is possible for two different Mapping Servers to advertise mappings that are conflicting and/or overlapping – the Mapping Client must handle this as well.

Note that at the time of writing this boot, there are discussions ongoing within the IETF for arriving at a consensus of handling of such conflict scenarios in a deterministic manner across implementations – refer to draft-ietf-spring-conflict-resolution. Based on the standardization of the conflict resolution mechanisms, the implementation in Cisco IOS XR may also get updated in the future for handling the conflicts on the part of Mapping Client function – there is no change expected on the conflict check for ensuring consistency during configuration on the part of Mapping Server function.

8.6.1 Overlapping and Conflicting Mapping Ranges Examples

The examples in this section use IPv4; equivalent rules exist for IPv6 mapping ranges. An example of non-conflicting, but overlapping set of prefix-to-SID mapping ranges: (1.1.1.1/32, 100, 10) and (1.1.1.6/32, 105, 5). The mapping ranges are expressed as a tuple (P/L, S, N), with P/L: first prefix and length in range; S: first SID index in range; and N: range size. The individual prefix-to-SID mapping entries are displayed in Table 8-1. The mapping entries of the first range are shown in the left columns, the mapping entries of the second range in the right columns.

Table 8-1: Non-conflicting, overlapping prefix-to-SID mappings

Mapping entry of range (1.1.1.1/32, 100, 10)		**Mapping entry of range (1.1.1.6/32, 105, 5)**	
prefix	**SID index**	**prefix**	**SID index**
1.1.1.1/32	100		
1.1.1.2/32	101		
1.1.1.3/32	102		
1.1.1.4/32	103		
1.1.1.5/32	104		
1.1.1.6/32	**105**	**1.1.1.6/32**	**105**
1.1.1.7/32	**106**	**1.1.1.7/32**	**106**
1.1.1.8/32	**107**	**1.1.1.8/32**	**107**
1.1.1.9/32	**108**	**1.1.1.9/32**	**108**
1.1.1.10/32	**109**	**1.1.1.10/32**	**109**

There is no conflict, the prefixes in both ranges are mapped to the same prefix-SID index, but some of the entries in the ranges are common. In such case, the Mapping Client only selects one of the ranges. The configuration verification in IOS XR ensures that no such overlapping entries can be configured in the Local Policy. See Example 8-4.

Example 8-4: Configuration error overlapping prefix-to-SID mappings

```
RP/0/0/CPU0:xrvr-1#configure
RP/0/0/CPU0:xrvr-1(config)#segment-routing
RP/0/0/CPU0:xrvr-1(config-sr)# mapping-server
RP/0/0/CPU0:xrvr-1(config-sr-ms)#  prefix-sid-map
RP/0/0/CPU0:xrvr-1(config-sr-ms-map)#   address-family ipv4
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#    1.1.1.1/32 100
range 10
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#    1.1.1.6/32 105
range 5
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#commit

% Failed to commit one or more configuration items during a
pseudo-atomic operation. All changes made have been reverted.
Please issue 'show configuration failed [inheritance]' from
this session to view the errors
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#show configuration
failed
!! SEMANTIC ERRORS: This configuration was rejected by
!! the system due to semantic errors. The individual
!! errors with each failed configuration command can be
!! found below.


segment-routing
 mapping-server
  prefix-sid-map
   address-family ipv4
    1.1.1.6/32 105 range 5
!!% Invalid argument: IP overlap
     1.1.1.6/32 .. 1.1.1.10/32
     Overlapping item is 1.1.1.1/32 100 10
   !
  !
 !
!
end
```

If different SID indexes are assigned to the same prefix, then there is a "prefix conflict". Prefix conflicts can only occur between mapping ranges of the same address-family and same prefix length. Consider the following set of prefix-to-SID mapping ranges: (1.1.1.1/32, 100, 10) and (1.1.1.6/32, 10, 5). Each individual prefix-to-SID mapping entry is displayed in Table 8-2. The mapping entries of the first range are shown in the left columns, the mapping entries of the second range in the right columns.

Table 8-2: Conflicting prefix-to-SID mappings: prefix conflict

Mapping entry of range (1.1.1.1/32, 100, 10)		**Mapping entry of range (1.1.1.6/32, 10, 5)**	
prefix	**SID index**	**prefix**	**SID index**
1.1.1.1/32	100		
1.1.1.2/32	101		
1.1.1.3/32	102		
1.1.1.4/32	103		
1.1.1.5/32	104		
1.1.1.6/32	**105 (!)**	**1.1.1.6/32**	**10 (!)**
1.1.1.7/32	**106 (!)**	**1.1.1.7/32**	**11 (!)**
1.1.1.8/32	**107 (!)**	**1.1.1.8/32**	**12 (!)**
1.1.1.9/32	**108 (!)**	**1.1.1.9/32**	**13 (!)**
1.1.1.10/32	**109 (!)**	**1.1.1.10/32**	**14 (!)**

The prefixes 1.1.1.n/32, with n=6…10 are mapped to two different SID indexes: 105…109 and 10…14. The Mapping Client could logically extract the entries of the prefix-to-SID mapping ranges that have no conflict, i.e. (1.1.1.1/32, 100, 5) in this example, but this would lead to additional implementation complexity on the client and is not considered. The configuration verification in IOS XR ensures that no such conflicting (and overlapping) ranges can be configured.

Example 8-5: Configuration error prefix conflict

```
RP/0/0/CPU0:xrvr-1#configure
RP/0/0/CPU0:xrvr-1(config)#segment-routing
RP/0/0/CPU0:xrvr-1(config-sr)# mapping-server
RP/0/0/CPU0:xrvr-1(config-sr-ms)#  prefix-sid-map
RP/0/0/CPU0:xrvr-1(config-sr-ms-map)#   address-family ipv4
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#    1.1.1.1/32 100
range 10
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#    1.1.1.6/32 10
range 5
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#commit

% Failed to commit one or more configuration items during a
pseudo-atomic operation. All changes made have been reverted.
Please issue 'show configuration failed [inheritance]' from
this session to view the errors
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#show configuration
failed
!! SEMANTIC ERRORS: This configuration was rejected by
!! the system due to semantic errors. The individual
!! errors with each failed configuration command can be
!! found below.


segment-routing
 mapping-server
  prefix-sid-map
   address-family ipv4
    1.1.1.6/32 10 range 5
!!% Invalid argument: IP overlap
     1.1.1.6/32 .. 1.1.1.10/32
     Overlapping item is 1.1.1.1/32 100 10
   !
  !
 !
!
end
```

If the same SID index is assigned to multiple prefixes, then there is a "SID index conflict". SID conflicts can occur between mapping entries of any address-family and prefix length. Consider the following set of prefix-to-

SID mapping ranges: (1.1.1.1/32, 100, 10) and (1.1.1.11/32, 100, 5). Each individual prefix-to-SID mapping entry is displayed in Table 8-3. The mapping entries of the first range are shown in the left columns, the mapping entries of the second range in the right columns.

Table 8-3: Conflicting prefix-to-SID mappings: SID index conflict

Mapping entry of range (1.1.1.1/32, 100, 10)		**Mapping entry of range (1.1.1.6/32, 105, 5)**	
prefix	**SID index**	**prefix**	**SID index**
1.1.1.1/32	**100 (!)**		
1.1.1.2/32	**101 (!)**		
1.1.1.3/32	**102 (!)**		
1.1.1.4/32	**103 (!)**		
1.1.1.5/32	**104 (!)**		
1.1.1.6/32	105		
1.1.1.7/32	106		
1.1.1.8/32	107		
1.1.1.9/32	108		
1.1.1.10/32	109		
		1.1.1.11/32	**100 (!)**
		1.1.1.12/32	**101 (!)**
		1.1.1.13/32	**102 (!)**
		1.1.1.14/32	**103 (!)**
		1.1.1.15/32	**104 (!)**

The same SID indexes: 100…104 have been assigned to the prefixes 1.1.1.n/32 and 1.1.1.m/32, with n=1…5 and m=11…15. Also here the Mapping Client could logically extract the part of the range in the prefix-to-SID mapping ranges that has no conflict, i.e. (1.1.1.6/32, 105, 5) in this example, but this is not considered due to the additional complexity. The configuration verification in IOS XR ensures that no such conflicting (and overlapping) ranges can be configured.

Example 8-6: Configuration error – SID index conflict

```
RP/0/0/CPU0:xrvr-1#configure
RP/0/0/CPU0:xrvr-1(config)#segment-routing
RP/0/0/CPU0:xrvr-1(config-sr)# mapping-server
RP/0/0/CPU0:xrvr-1(config-sr-ms)#  prefix-sid-map
RP/0/0/CPU0:xrvr-1(config-sr-ms-map)#   address-family ipv4
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#    1.1.1.1/32 100 range 10
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#    1.1.1.11/32 100 range 5
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#commit

% Failed to commit one or more configuration items during a pseudo-atomic operation. All changes made have been reverted. Please issue 'show configuration failed [inheritance]' from this session to view the errors
RP/0/0/CPU0:xrvr-1(config-sr-ms-map-af)#show configuration failed
!! SEMANTIC ERRORS: This configuration was rejected by
!! the system due to semantic errors. The individual
!! errors with each failed configuration command can be
!! found below.


segment-routing
 mapping-server
  prefix-sid-map
   address-family ipv4
    1.1.1.11/32 100 range 5
!!% Invalid argument: SID overlap:
      100 .. 104
```

```
        Overlapping item is 1.1.1.1/32 100 10
      !
    !
  !
!
end
```

8.6.2 Selecting Mapping Ranges from Different Mapping Servers

For redundancy purposes, multiple Mapping Servers should exist in the network. It is expected that these Mapping Servers all advertise the same prefix-to-SID mappings, but this may not always be the case. Different Mapping Servers can advertise different or even conflicting prefix-to-SID mappings. And Mapping Server advertised prefix-to-SID mappings can also conflict with other Prefix-SID advertisements. If a Mapping Client receives conflicting mapping entries, it cannot know which ones are correct and which ones are wrong. IETF draft-ietf-spring-conflict-resolution describes different strategies to cope with such conflicts. A first strategy could be to ignore all conflicting entries. Another strategy could be to select and use one of the conflicting entries by applying a set of rules. The latter approach is selected in the Cisco IOS XR implementation. To ensure consistent forwarding entries, the rules that are used must be such that they result in the same set of mapping (Local Mapping Policy) on each Mapping Client in the network.

If a Mapping Client receives two or more overlapping mapping ranges, it selects one of the overlapping ranges based on the following preference rules. These rules are applied sequentially on the set of mapping ranges until a rule results in a single range; this range is then selected as the preferred mapping range.

1. Largest router-id (OSPF) or system-id (ISIS) is preferred
2. Smallest area id (OSPF) or level (ISIS) is preferred
3. IPv4 range is preferred over IPv6 range
4. Smallest prefix length is preferred
5. Smallest IP address is preferred
6. Smallest SID index is preferred
7. Smallest range is preferred
8. First received range is preferred

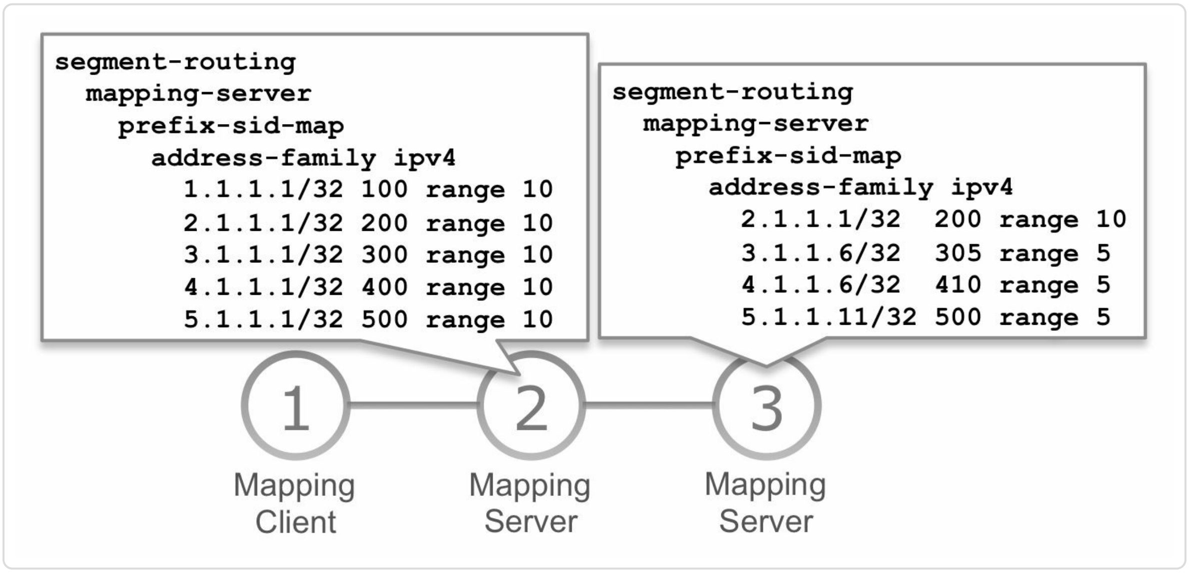
The Mapping Client inserts the preferred mapping range in the Active Policy, and the other, non-selected mapping ranges, in the Backup Policy.

The rules 3 to 8 are only used if an individual Mapping Server advertises overlapping mapping ranges or when an ABR propagates overlapping mapping ranges. The former would normally not be observed for a Cisco IOS XR Mapping Server since only non-conflicting, non-overlapping mapping ranges are accepted when configuring the Local Mapping Policy. [2]

The current implemented preference rules guarantee a consistent Active Mapping Policy for all Mapping Clients in a single area. However, they do not guarantee consistent Active Policies on all Mapping Clients in the network if mapping advertisements are propagated between areas or levels. When the ABR or L1L2 node propagates the prefix-to-SID mappings, it (re-)originates the mappings with its own router-id or system-id. This affects the prefix-to-SID mapping selection since the router-id or system-id is used in the first preference rule. Since the system-id or router-id of the propagated mapping advertisement changed, a prefix-to-SID mapping that is preferred in one area, may not be preferred in another area or the other way around.

IETF draft-ietf-spring-conflict-resolution specifies rules that ensure a consistent selection of prefix-to-SID mappings on all Mapping Clients and as mentioned previously not all of its rules are implemented in Cisco IOS XR pending consensus at the time of writing this book.

The following example illustrates the selection of Prefix-to-SID mappings. Figure 8-2 shows a topology with two Mapping Servers, Node2 and Node3, and a Mapping Client, Node1. The Mapping Servers advertise one common mapping range and three mapping ranges that overlap and/or conflict with the other Mapping Server's mapping ranges. The overlapping and conflicting mapping ranges are equivalent to the ranges used in section 8.6.1, "Overlapping and conflicting mapping ranges – examples". See that section for more details.



```
segment-routing
  mapping-server
    prefix-sid-map
      address-family ipv4
        1.1.1.1/32 100 range 10
        2.1.1.1/32 200 range 10
        3.1.1.1/32 300 range 10
        4.1.1.1/32 400 range 10
        5.1.1.1/32 500 range 10
```

```
segment-routing
  mapping-server
    prefix-sid-map
      address-family ipv4
        2.1.1.1/32  200 range 10
        3.1.1.6/32  305 range 5
        4.1.1.6/32  410 range 5
        5.1.1.11/32 500 range 5
```

Figure 8-2: network topology with two Mapping Servers

Table 8-4 shows the advertised mapping ranges side-by-side. The prefix-to-SID mappings are written as tuples (P/L, S, N). With P/L: first prefix; S:

first SID index; N: range size.

Table 8-4: Mapping Servers prefix-to-SID mappings

Advertised by Node2 (router- id 1.1.1.2)	Advertised by Node3 (router-id 1.1.1.3)	Comment	(1.1.1.1/32, 100, 10)
none	Non-overlapping	(2.1.1.1/32, 200, 10)	**(2.1.1.1/32, 200, 10)**
Identical	(3.1.1.1/32, 300, 10)	**(3.1.1.6/32, 305, 5)**	Overlapping, non-conflicting
(4.1.1.1/32, 400, 10)	**(4.1.1.6/32, 410, 5)**	Prefix conflict	(5.1.1.1/32, 500, 10)

For the overlapping mapping ranges (rows 2 to 5 in Table 8-4), the Mapping Client prefers the ranges advertised by the Mapping Server with the highest router-id or system-id, Node3 in this example. The Mapping Client also selected the non-overlapping range (row 1 in Table 8-4), which is only advertised by Node2. The preferred mapping ranges are highlighted in Table 8-4. The Local Mapping Policy configurations on Node2 and Node3 are displayed in Example 8-7 and Example 8-8.

Example 8-7: Local Mapping configuration Node2

```
segment-routing
 mapping-server
  prefix-sid-map
   address-family ipv4
    1.1.1.1/32 100 range 10
    2.1.1.1/32 200 range 10
    3.1.1.1/32 300 range 10
    4.1.1.1/32 400 range 10
    5.1.1.1/32 500 range 10
   !
  !
 !
!
```

Example 8-8: Local Mapping configuration Node3

```
segment-routing
 mapping-server
  prefix-sid-map
   address-family ipv4
    2.1.1.1/32 200 range 10
    3.1.1.6/32 305 range 5
    4.1.1.6/32 410 range 5
    5.1.1.11/32 500 range 5
   !
  !
 !
!
```

The Local Mapping Policy is independent of the IGP. The Active Policy and Backup Policy however, are derived per IGP. These Mapping Policies can be verified in IGP by using the command `show isis|ospf segment-routing prefix-sid-map active-policy|backup-policy`. See the output for OSPF on Mapping Client Node1 in Example 8-9 and the detailed output in Example 8-10.

Example 8-9: Active Mapping Policy

```
RP/0/0/CPU0:xrvr-1#show ospf segment-routing prefix-sid-map
active-policy

        SRMS active policy for Process ID 1

Prefix                SID Index    Range        Flags
2.1.1.1/32            200          10
3.1.1.6/32            305          5
4.1.1.6/32            410          5
5.1.1.11/32           500          5
1.1.1.1/32            100          10

Number of mapping entries: 5
```

Example 8-10: Detailed Active Mapping Policy

```
RP/0/0/CPU0:xrvr-1#show ospf segment-routing prefix-sid-map
active-policy detail

        SRMS active policy for Process ID 1

Prefix
2.1.1.1/32
    Source:         Remote
    Router ID:      1.1.1.3
    Area ID:        1
    SID Index:      200
    Range:          10
    Last Prefix:    2.1.1.10/32
    Last SID Index: 209
    Flags:
3.1.1.6/32
    Source:         Remote
    Router ID:      1.1.1.3
    Area ID:        1
    SID Index:      305
    Range:          5
    Last Prefix:    3.1.1.10/32
    Last SID Index: 309
    Flags:
4.1.1.6/32
    Source:         Remote
    Router ID:      1.1.1.3
    Area ID:        1
    SID Index:      410
    Range:          5
    Last Prefix:    4.1.1.10/32
    Last SID Index: 414
    Flags:
5.1.1.11/32
    Source:         Remote
    Router ID:      1.1.1.3
    Area ID:        1
    SID Index:      500
    Range:          5
    Last Prefix:    5.1.1.15/32
    Last SID Index: 504
    Flags:
1.1.1.1/32
```

```
    Source:          Remote
    Router ID:       1.1.1.2
    Area ID:         1
    SID Index:       100
    Range:           10
    Last Prefix:     1.1.1.10/32
    Last SID Index:  109
    Flags:

Number of mapping entries: 5
```

From the overlapping mapping ranges, the Mapping Client on Node1 preferred the ones advertised by the node with the highest router-id, Node3. Node2 advertises the last mapping range shown in the Active Mapping Policy, since Node3 does not advertise a mapping range that overlaps with this one.

The non-preferred overlapping mapping ranges are added to the Backup Mapping Policy. See Example 8-11. In this example, the Backup Mapping Policy contains the mapping ranges that Node2 advertises.

Example 8-11: Backup Mapping Policy

```
RP/0/0/CPU0:xrvr-1#show ospf segment-routing prefix-sid-map
backup-policy

        SRMS backup policy for Process ID 1

Prefix                    SID Index     Range         Flags
2.1.1.1/32                200           10
3.1.1.1/32                300           10
4.1.1.1/32                400           10
5.1.1.1/32                500           10

Number of mapping entries: 4
```

8.6.3 Mapping Range Conflict with “Native”

Prefix-SID

As a reminder, IGPs receive Prefix-SID assignments via local configuration on the router as well via IGP Advertisement of Prefix-SIDs as described in chapter 5, “Segment Routing IGP Control Plane”. These assignments provide mapping of a prefix to its SID as well but the difference is that these are coming from nodes where they has likely been explicitly configured. For sake of simplicity, we describe them as “native” Prefix-SID mapping and they could be considered more authoritative as compared to the SID assignments coming from a source such as the Mapping Server. This is one viewpoint and should not be taken as a consensus.

Now it is possible for the IGP to get a Prefix-SID mapping from the Active Mapping Policy of the Mapping Client function for a prefix for which it also gets a “native” Prefix-SID assignment. If the SID value is the same then there is no issue, but if they are different then we have a conflict scenario. At the time of writing this book, the Cisco IOS-XR IGP implementations prefers the “native” Prefix-SID over the mapping received via the SRMS – i.e. the implementation tries to find a SID mapping for a prefix in the active mapping policy only when it has not received one “natively”. As mentioned previously, this behavior is also the subject of draft-ietf-spring-conflict-resolution that has not reached a consensus at the time of writing this book.

HIGHLIGHT: Making Changes to SRMS Mappings

Once Mapping Servers are deployed with certain mapping ranges configuration and used actively in a deployment, extra care must be taken when making any configuration changes to it.

Re-emphasize the concept of conflicting and/or overlapping ranges and the impact when conflicts occur.

It is best to provision mappings on multiple Mapping Servers concurrently and without errors with use of centralized provisioning tools like Cisco Network Service Orchestrator.

When mapping range(s) configurations are changed in a manner that there are conflicts, it is possible that some routers would lose their prefix segments and in this case the forwarding switches over to native IP or LDP.

Traffic loss and service impact could happen if different router implementations interpret and perform conflict resolution differently as this could lead to inconsistent forwarding state across routers; monitor how the consensus is evolving at IETF for draft-ietf-spring-conflict-resolution.

8.7 IGP Configuration for SRMS

8.7.1 Mapping Server

To act as a Mapping Server, the advertisement of the locally configured prefix-to-SID mappings (the Local Mapping Policy) must be enabled in the IGP. Configure `segment-routing prefix-sid-map advertise-local` under the IGP to enable the advertisement. See Example 8-12 for ISIS and Example 8-13 for OSPF.

Example 8-12: ISIS Mapping Server configuration

```
router isis 1
 address-family ipv4 unicast
  segment-routing prefix-sid-map advertise-local
 !
 address-family ipv6 unicast
  segment-routing prefix-sid-map advertise-local
!
```

Example 8-13: OSPF Mapping Server configuration

```
router ospf 1
 segment-routing prefix-sid-map advertise-local
!
```

Only one Local Mapping Policy can be configured on a given node. If multiple IGP instances on a node have the Mapping Server functionality enabled, they all use and advertise the same Local Mapping Policy.

If the Mapping Server function is enabled on an L1L2 IS-IS instance, then the Local Mapping Policy is advertised in all levels. If the Mapping Server function is enabled on an OSPF ABR, then the Local Mapping Policy is advertised in all Segment Routing enabled areas of that OSPF instance.

The `segment-routing prefix-sid-map advertise-local` command for OSPF has instance scope: it applies to all areas of an instance. Mapping Server advertisements need not be enabled/disabled per area since the advertisements are in any case flooded by IGPs across the areas/levels However, if the actual prefixes are not available in the area (e.g. due to filtering or summarization) then just the Prefix-SID mapping advertisements would have no effect.

Example 8-14 shows an example of a Mapping Server configured to advertise a set of IPv4 and IPv6 prefix-to-SID mappings in ISIS.

Example 8-14: Local Mapping Policy and ISIS Mapping Server configuration

```
segment-routing
 mapping-server
  prefix-sid-map
   address-family ipv4
    1.1.1.0/32 10 range 40
    2.1.1.0/24 100 range 60
   !
   address-family ipv6
    2001::1:1:1:0/128 2010 range 40
    2001::2:1:1:0/112 2100 range 60
!
router isis 1
 address-family ipv4 unicast
  segment-routing prefix-sid-map advertise-local
```

Two IPv4 prefix-to-SID mapping ranges are configured in this example. The first one is a range of 40 mappings for a prefix range starting at 1.1.1.0/32, with the Prefix-SID index range starting at index 10. As a result of this configuration, prefix 1.1.1.0/32 is mapped to Prefix-SID index 10, 1.1.1.1/32 is mapped to Prefix-SID index 11, and so on until prefix 1.1.1.39/32 which is mapped to Prefix-SID index 49. All prefixes have the same prefix length. See Table 8-5.

Table 8-5: IPv4 prefix to Prefix-SID index mappings

Prefix	Prefix-SID index
1.1.1.0/32	10
1.1.1.1/32	11
…	…
1.1.1.39/32	49

Typically, Prefix-SID indexes are associated with host prefixes, but they can be associated with prefixes of other lengths. To illustrate the use of a prefix length different from the host prefix length, the second mapping range in the example shows a prefix-to-SID mapping range with a prefix of length 24. All prefixes in the range will have the same prefix length 24 and the prefixes in the range are incremented based on their prefix length. In the example of a /24 prefix, the third digit in the prefix is incremented.

As a result of this configuration, prefix 2.1.1.0/24 is mapped to Prefix-SID index 100, 2.1.2.0/24 is mapped to Prefix-SID index 101, and so on until prefix 2.1.60.0/24 which is mapped to Prefix-SID index 159. All prefixes have the same prefix length 24. See Table 8-6.

Table 8-6: IPv4 prefix to Prefix-SID index mappings, non-host prefixes

Prefix	Prefix-SID index
2.1.1.0/24	100
2.1.1.1/24	101
…	…
2.1.1.49/24	159

8.7.2 Mapping Client

A typical Mapping Client configuration does not have any Local Mapping Policy configured. And it is not configured to advertise local prefix-to-SID mappings in the IGP.

The Mapping Client functionality receives, parses, and processes Mapping Server advertisements that are received by the IGP. The Mapping Client functionality is enabled by default on all Segment Routing nodes. If required, the Mapping Client functionality can be disabled by configuring `segment-routing prefix-sid-map receive disable` under the IGP. See Example 8-15 and Example 8-16.

Example 8-15: disable Mapping Client function in ISIS

```
router isis 1
 address-family ipv4 unicast
  segment-routing prefix-sid-map receive disable
 !
 address-family ipv6 unicast
  segment-routing prefix-sid-map receive disable
 !
!
```

Example 8-16: disable Mapping Client function in OSPF

```
router ospf 1
 segment-routing prefix-sid-map receive disable
!
```

8.7.3 Mapping Server and Mapping Client

As seen previously, it is possible for the same node to host both the Mapping Server and Client function. Table 8-7 describes the Mapping Server and Mapping Client behavior resulting from the different `prefix-sid-map` configuration options under the IGP configuration. The first column indicates the configuration options. The middle column, "Advertise Local Policy?", indicates if the prefix-to-SID mappings of the Local Mapping Policy are advertised with the configuration in the first column. The third column, "Compute Active Policy", indicates which mappings are used when deriving the Active Mapping Policy. "Local mappings" are the prefix-to-SID mappings of the Local Mapping Policy, and "Remote mappings" are the prefix-to-SID mappings received from remote Mapping Servers.

Table 8-7: Mapping Server and Mapping Client configuration options

Configuration option	**Advertise Local Policy?**	**Compute Active Policy**
`receive` (default)	No	Ignore local mappings Use remote mappings
`advertise-local + receive disable`	Yes	Use local mappings Ignore remote mappings
`advertise-local + receive`	Yes	Use local mappings Use remote mappings

From Table 8-7:

- If only `segment-routing prefix-sid-map receive` is configured (this is the default and is not displayed in the configuration), then the Local Mapping Policy, if configured, is not advertised in the IGP. With this configuration the locally configured mapping policy is ignored and only prefix-to-SID mappings received from remote Mapping Servers are used to compute the Active Policy. This is a typical Mapping Client configuration.
- If `segment-routing prefix-sid-map advertise-local` and `segment-routing prefix-sid-map receive`

`disable` are configured, then the Local Mapping Policy is advertised by the IGP. With this configuration the prefix-to-SID mappings received from remote Mapping Servers are ignored and only the prefix-to-SID mappings in the Local Mapping Policy are used to compute the Active Policy. Practically this is a combination which should never be used.

- If both `segment-routing prefix-sid-map receive` and `segment-routing prefix-sid-map advertise-local` are configured, then the Local Mapping Policy is advertised in IGP and both local and remote prefix-to-SID mappings are used to compute the Active Policy. This is a typical Mapping Server configuration.

8.8 Mapping Server Advertisements in IS-IS

The IS-IS Mapping Server advertises prefix-to-SID mappings using SID/Label Binding TLVs, which are top level TLVs in the ISIS LSP. The Prefix-SID sub-TLV is included as a sub-TLV of this SID/Label Binding TLV. Each prefix-to-SID mapping range (one line in the Local Mapping configuration) is encoded in a separate SID/Label Binding TLV.

The IETF ISIS SR extensions draft[3] describes three applications of the SID/Label Binding TLV:

- Advertise a SID/Label binding to a FEC along with at least a single 'nexthop style' anchor.
- Advertise SID/Label Bindings and their associated Primary and Backup paths.
- Advertise prefixes to SID/Label mappings for the "Mapping Server" functionality.

Only the Mapping Server application is covered in this book. Details of the other applications can be found in the IETF draft.

Figure 8-3 shows the format of the SID/Label Binding TLV.

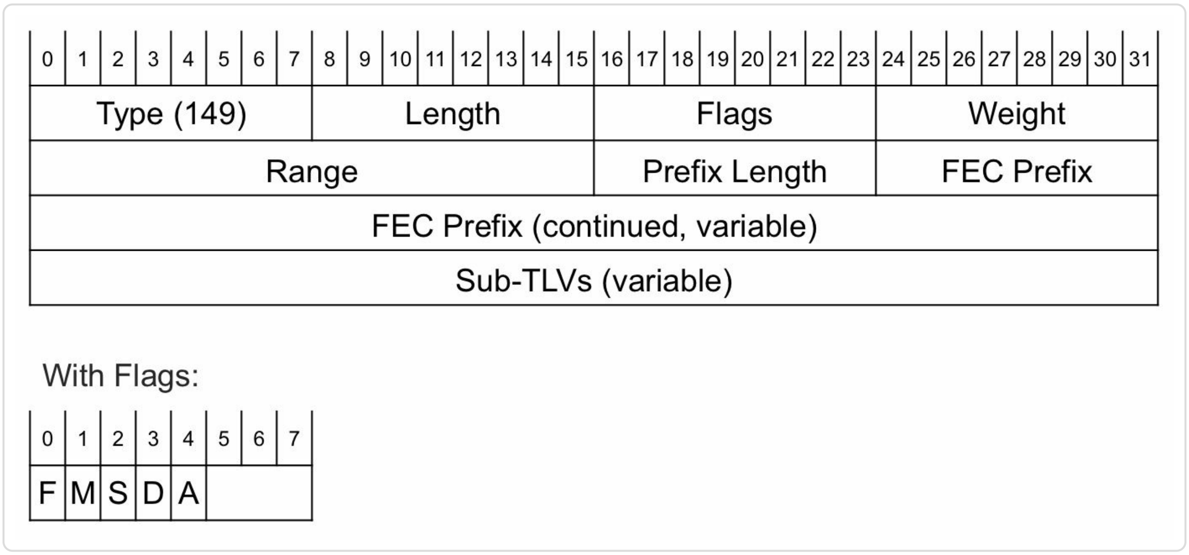


Figure 8-3: ISIS SID/Label Binding TLV format

The SID/Label Binding TLV contains the following fields:

- Flags:
  - F (Address-Family): if unset, then the FEC Prefix is an IPv4 prefix; if set, then the FEC Prefix is an IPv6 prefix
  - M (Mirror Context): set if the SID/path corresponds to a mirrored context. See ietf-spring-segment-routing for more details – IOS XR: always unset
  - S (Scope): if unset, then this TLV must not be propagated between levels; if set, then this TLV may be flooded across the entire domain. Usage of this flag is equivalent to the S flag in the Router Capability TLV – IOS XR: always unset
  - D (Down): set if this TLV is propagated from Level-2 to Level-1, unset otherwise. If set, then don't propagate from Level-1 to Level-2. Used to prevent unnecessary flooding of this TLV. Usage of this flag is equivalent to the D flag in the Router Capability TLV – IOS XR: always unset

    - A (Attached): set if the prefixes in the TLV are known to be attached to their originators. The A-flag must be unset if this TLV is leaked to another level/area. – IOS XR default: unset

- Weight field: The value represents the weight of the path for the purpose of load balancing – IOS XR: always 0
- Range: number of prefix-to-SID mapping entries in the mapping range
- Prefix length and FEC prefix: the first prefix in the mapping range
- sub-TLV: The Prefix-SID sub-TLV which indicates the first Prefix-SID index of the prefix-to-SID mapping range

The Prefix-SID sub-TLV was covered earlier in this book, in the SR IGP control plane chapter 5. In that section the Prefix-SID sub-TLV was used to advertise a "native" Prefix-SID, a Prefix-SID attached to an IP Reachability TLV. The Mapping Server application uses the same sub-TLV, but there is a difference in interpretation of the flags in that sub-TLV. The format of the Prefix-SID sub-TLV is repeated here in Figure 8-4, with the Flags field adapted to its use here. The Prefix-SID sub-TLV used in the SID/Label Binding TLV may only use the N-flag.

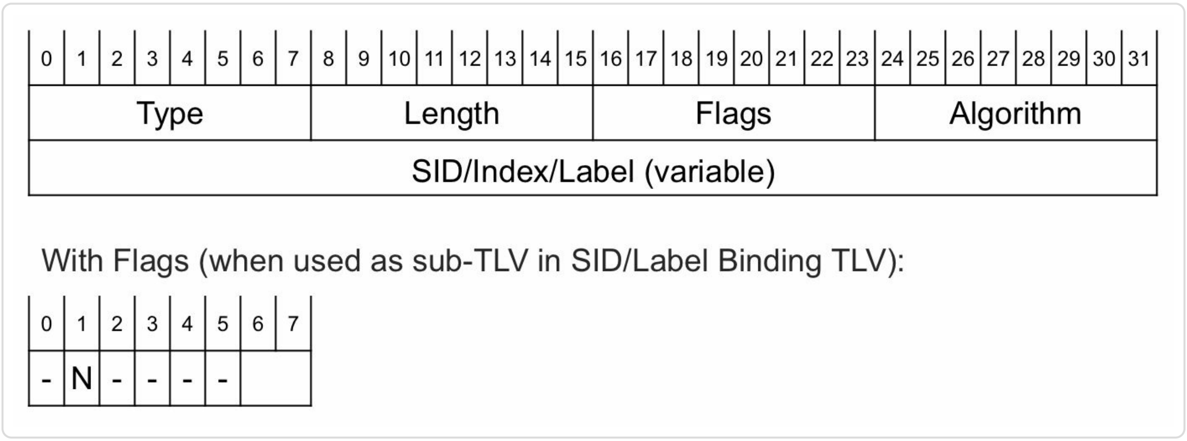


Figure 8-4: Prefix-SID sub-TLV flags field

The Flags in the Prefix-SID sub-TLV that are used when included in the SID/Label Binding TLV:

- N (Node-SID): set if the Prefix-SID is a Node-SID, i.e. identifies the node – always unset in Cisco IOS XR

The other flags in this sub-TLV are not used for the Mapping Server application and are ignored at reception.

The Algorithm field in the Prefix-SID sub-TLV is set to 0 (SPF) in Cisco IOS XR.

The Mapping Server advertisements can be verified in the IS-IS link-state database. Example 8-17 shows the output of `show isis database verbose` for an ISIS LSP advertised by a Mapping Server. The Mapping Server has the Local Mapping Policy configured as in Example 8-14. Each configured Local Mapping Policy entry is advertised as a separate TLV in the Mapping Server's ISIS LSP.

Example 8-17: ISIS Mapping Server advertisements

```
RP/0/0/CPU0:xrvr-5#show isis database verbose xrvr-5

IS-IS 1 (Level-2) Link State Database
LSPID                     LSP Seq Num  LSP Checksum  LSP
Holdtime  ATT/P/OL
xrvr-5.00-00            * 0x000000d9   0x9c55
1100            0/0/0
  Area Address:   49.0002
  NLPID:          0xcc
  NLPID:          0x8e
  MT:             Standard (IPv4 Unicast)
  MT:             IPv6
Unicast                                0/0/0
  Hostname:       xrvr-5
  IP Address:     1.1.1.5
  IPv6 Address:   2001::1:1:1:5
  Router Cap:     1.1.1.5, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
<...>
  SID Binding:    1.1.1.0/32 F:0 M:0 S:0 D:0 A:0 Weight:0 Range:40
    SID: Start:10, Algorithm:0, R:0 N:0 P:0 E:0 V:0 L:0
  SID Binding:    2.1.1.0/24 F:0 M:0 S:0 D:0 A:0 Weight:0 Range:60
    SID: Start:100, Algorithm:0, R:0 N:0 P:0 E:0 V:0 L:0
  SID Binding:    2001::1:1:1:0/128 F:1 M:0 S:0 D:0 A:0 Weight:0 Range:40
    SID: Start:2010, Algorithm:0, R:0 N:0 P:0 E:0 V:0 L:0
  SID Binding:    2001::2:1:1:0/112 F:1 M:0 S:0 D:0 A:0 Weight:0 Range:60
    SID: Start:2100, Algorithm:0, R:0 N:0 P:0 E:0 V:0 L:0

 Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

8.9 Mapping Server Advertisements inOSPF

An OSPF Mapping Server advertises prefix-to-SID mappings using Extended Prefix Opaque LSAs (Opaque-Type 7), the same type of LSA as for the “native” Prefix-SID advertisements. The prefix-to-SID mappings are encoded in an OSPFv2 Extended Prefix Range TLV, a new top level TLV of the Extended Prefix LSA. The OSPFv2 Extended Prefix LSA can also carry the Extended Prefix TLV. But contrary to the Extended Prefix TLV, which is associated with a single prefix, the Extended Prefix Range TLV is for specifying attributes for a range and hence they are signaled separately. In Cisco IOS XR each configured Local Mapping Policy entry is advertised in a separate LSA containing a single Extended Prefix Range TLV to make it simpler to signal changes to the mapping ranges.

Figure 8-5 shows the format of the Extended Prefix Range TLV.

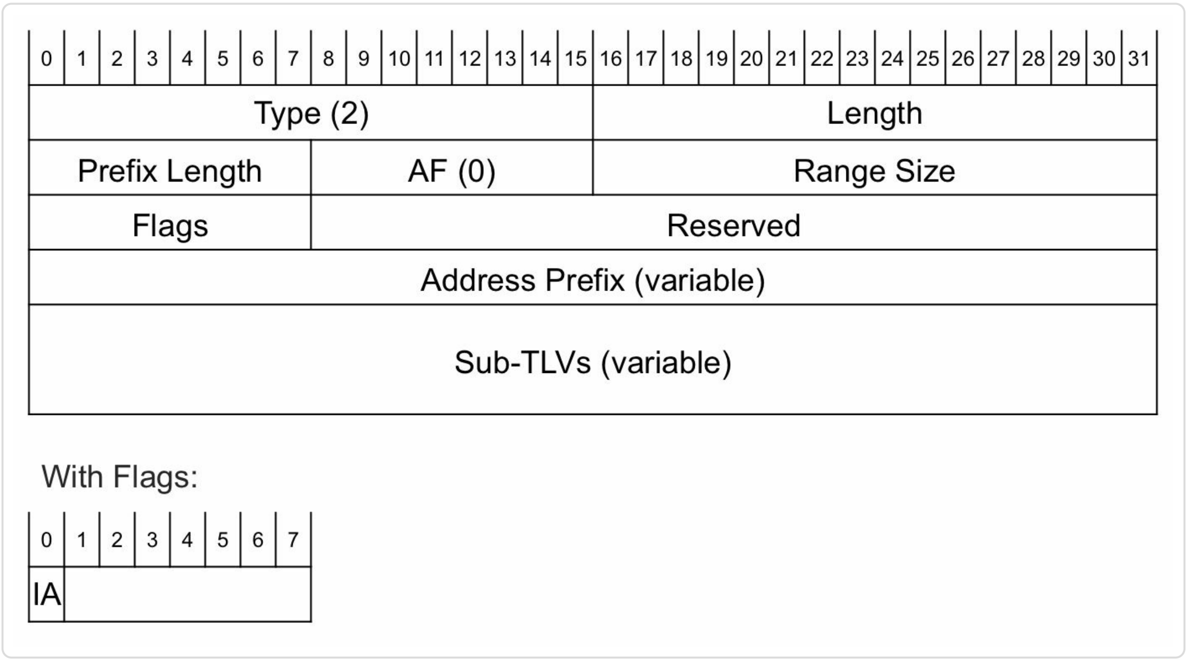


Figure 8-5: OSPFv2 Extended Prefix Range TLV

The OSPFv2 Extended Prefix Range TLV contains the following fields:

- Prefix length: length of the prefix
- AF (Address Family): 0 – IPv4 unicast
- Range size: number of prefix-to-SID mapping entries in the mapping range
- Flags:
  - IA (Inter-Area): set if the advertisement is propagated between areas. This flag is used to prevent looping of Mapping Server advertisements between areas. An ABR does not propagate a Mapping Server advertisement that has been received from a non-backbone area and has the IA-flag set.
- Address Prefix: the first prefix in the prefix range
- sub-TLVs: different sub-TLVs can be included, such as the Prefix-SID sub-TLV

The Prefix SID sub-TLV is already described in chapter 5, “Segment Routing IGP Control Plane” and its format is repeated in Figure 8-6. If this sub-TLV is included in a Mapping Server advertisement, the treatment of some of the fields is different compared to its use in the Extended Prefix TLV to advertise a “native” Prefix-SID.

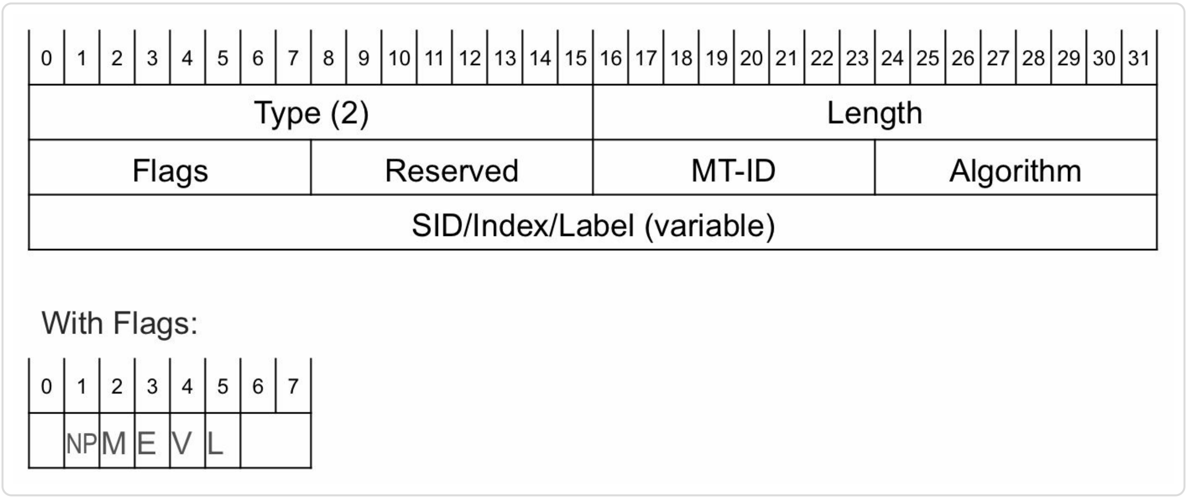


Figure 8-6: OSPFv2 Prefix-SID sub-TLV format

The OSPFv2 Prefix-SID sub-TLV contains the following fields:

- Flags:
  - NP (no-PHP): ignored if M-flag is set, i.e. for Mapping Server advertisement
  - M (Mapping Server): set since it is advertised from a Mapping Server
  - E (Explicit-Null): ignored if M-flag set, i.e. for Mapping Server advertisement
  - V (Value): set if prefix-SID carries an absolute value, unset if the Prefix-SID carries an index – Cisco IOS XR: always unset
  - L (Local/Global): set if prefix-SID has local significance, unset if prefix-SID has global significance – Cisco IOS XR: always unset
- MT-ID (Multi-Topology ID (as defined in IETF RFC 4915)) – IOS XR: always default topology (0)
- Algorithm: the algorithm the Prefix-SID is associated with. – IOS XR: set to 0

- SID/Index/Label: if V-flag and L-flag unset: the Prefix-SID index. When this Prefix-SID sub-TLV is advertised in an Extended Prefix Range TLV (i.e. a Mapping Server advertisement), then the Prefix-SID value is interpreted as a starting SID index value.

Use the command `show ospf database opaque-area` to verify prefix-to-SID mapping advertisements for OSPF, as in Example 8-18. Remember these Mapping Server advertisements are carried in Extended Prefix Opaque LSAs (Opaque-type 7). The Opaque LSAs used for Mapping Server advertisements have Area-scope (type 10 LSA) in Cisco IOS XR. The Mapping Server advertises each configured Local Mapping Policy range in a separate LSA.

Example 8-18: OSPFv2 Mapping Server advertisements

```
RP/0/0/CPU0:xrvr-5#show ospf database opaque-area 7.0.0.3
self-originate


            OSPF Router with ID (1.1.1.5) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area
0)

  LS age: 51
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.3
  Opaque Type: 7
  Opaque ID: 3
  Advertising Router: 1.1.1.5
  LS Seq Number: 80000001
  Checksum: 0xa790
  Length: 48

    Extended Prefix Range TLV: Length: 24
      AF         : 0
      Prefix     : 1.1.1.0/32
```

```
       Range Size: 40
       Flags     : 0x0            !! IA:0

       SID sub-TLV: Length: 8
         Flags     : 0x60         !! (NA); NP:1; M:1; E:0;
V:0; L:0
         MTID      : 0
         Algo      : 0
         SID Index : 10

```

Note: The NP-flag (PHP-off) is set in this example. This is based on an earlier version (-05) of the IETF draft-ietf-ospf-segment-routing-extensions. Current IETF draft version (-08) specifies that the NP-flag and E-flag must be ignored for Mapping Server advertisements, i.e. if the M-flag is set.

8.10 Mapping Server in Multi-Area/LevelNetwork

8.10.1 ISIS Inter-Level SRMS Advertisements

At the time of writing this book, ISIS in Cisco IOS XR does not propagate the Mapping Server advertisements between levels. This means that for multi-level IS-IS networks, a Mapping Server per IS-IS area/level is currently required.

8.10.2 OSPF Inter-Area SRMS Advertisements

The Mapping Server prefix-to-SID mapping is advertised in an OSPFv2 Extended Prefix LSA with an OSPFv2 Extended Prefix Range TLV. One Extended Prefix LSA is generated for each local prefix-to-SID mapping range. These Extended Prefix Opaque LSAs for Mapping Server advertisements have area flooding scope (Type 10 LSA) in Cisco IOS XR, which means that an ABR must propagate the Extended Prefix Range TLVs between areas to achieve network-wide distribution.

If the Mapping Server is an Area Border Router (ABR) then the Extended Prefix LSA with Extended Prefix Range TLV is advertised in each connected area where Segment Routing is enabled.

When an Area Border Router receives an Extended Prefix Range TLV in a Mapping Server advertisement, then it propagates it to all connected areas except the one from which the Mapping Server advertisement was received. To prevent redundant flooding of Extended Prefix Range TLV between areas, the following logic is used:

- ABR sets IA-flag (Inter-Area) in the Extended Prefix Range TLV when propagating a prefix-to-SID mapping advertisement to another area. A Prefix-to-SID mapping in an Extended Prefix Range TLV with the IA-flag set is called an “inter-area mapping”.
- ABR never propagates an Extended Prefix Range TLV that was received from a non-backbone area and has the IA-flag (Inter-Area) set
- If another reachable node advertises an Extended Prefix Range TLV as an intra-area (IA-flag=0) mapping in an area, then ABR does not originate that same Extended Prefix Range TLV as inter-area (IA-flag=1) mapping advertisement in that area

Note that the propagation of Prefix-to-SID mapping advertisements is independent from the selection of a preferred Prefix-to-SID mapping from a set of overlapping mappings. OSPF propagates all mapping range advertisements, but prevents redundant flooding of the same mapping range (the same Extended Prefix Range TLV).

Figure 8-7 and Figure 8-8 illustrate the mapping advertisement propagation and redundant flooding prevention mechanism. The network topology consists of three areas, with two ABRs between the backbone area (0) and each of the two other areas (1 and 2). Node5 is a mapping Server and advertises a prefix-to-SID mapping range into Area 1: (N5, M1, IA=0), where N5 is the advertising node, M1 is the prefix-to-SID mapping range, and IA=0 is the inter-area flag. ABRs Node1 and Node2 both propagate the mapping range to Area 0 and set the IA-flag on the propagated mapping range. Two instances of the mapping range M1 advertisement are present in Area 0: (N1, M1, IA=1) from Node1, and (N2, M1, IA=1) from Node2. ABRs Node3 and Node4 do not see an intra-area advertisement of mapping range M1 in Area 2, hence they both propagate the mapping range M1 advertisement to Area 2: (N3, M1, IA=1) from Node3, and (N4, M1, IA=1) from Node4. Redundant flooding is prevented. For example, ABR Node4 does not propagate the mapping range (N3, M1, IA=1) to Area 2 since an inter-area mapping (IA-flag=1) advertisement is never propagated from a non-backbone area.

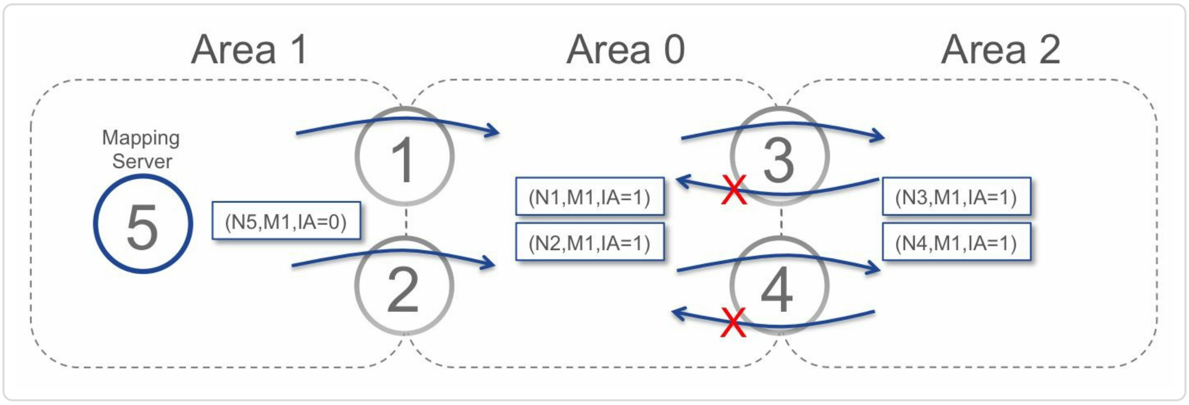


Figure 8-7: Mapping Server advertisements propagation

Now Mapping Server Node6 is added in Area 2, see Figure 8-8. Both Node5 and Node6 advertise the same mapping range M2. ABRs Node1 and Node2 both propagate the mapping range M2 advertisement to Area 0. However, ABRs Node3 and Node4 do not propagate the mapping range M2 advertisement to Area 2 since Node6 already advertises the mapping range M2 in Area 2 as intra-area advertisement (IA=0). Similarly, ABRs Node1 and Node2 do not propagate the inter-area mapping ranges advertised by Node3 and Node4 in Area 0, into Area 1 because of the intra-area mapping advertisement of Node5 in Area 1.

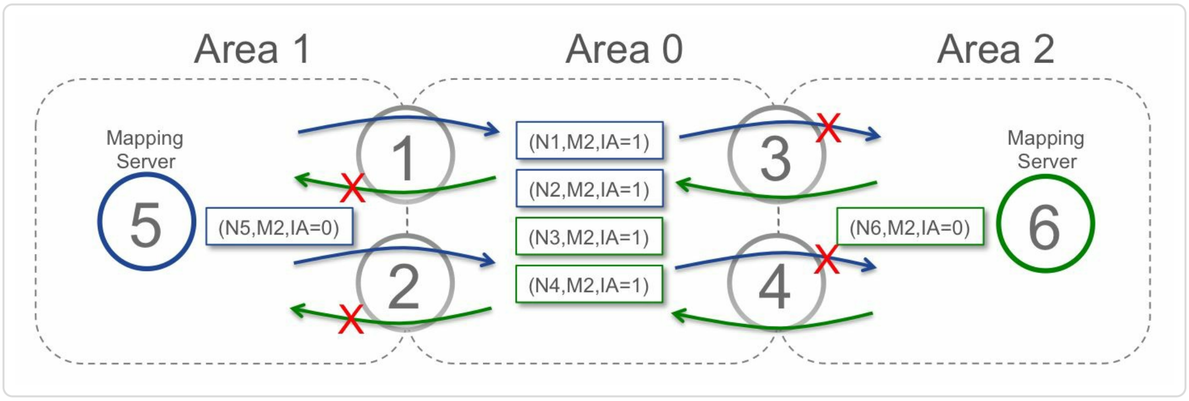


Figure 8-8: Mapping Server advertisements propagation

Let us look closer at how the propagation of Mapping Server advertisements happens in OSPF. For this illustration, with router configurations and show commands, the same topology as in the OSPF inter-area section is used (see chapter 5). The topology is repeated here in Figure 8-9. It is a chain of 7 nodes. Node1, Node2, and Node3 are in area 1. Node3, Node4, and Node5 are in area 0. Node5, Node6, and Node7 are in area 2.

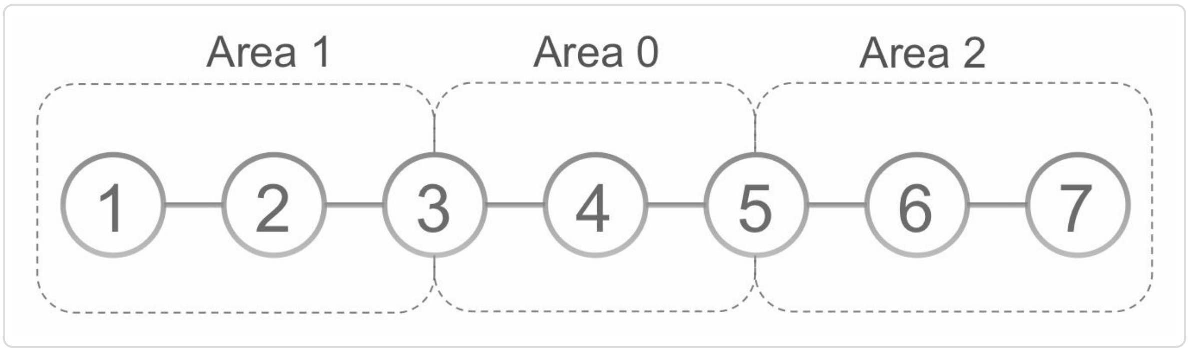


Figure 8-9: inter-area network topology

Node1 is configured as Mapping Server with the Local Mapping Policy configuration shown in Example 8-19.

Example 8-19: Mapping Server Local Policy configuration

```
segment-routing
```

```
 mapping-server
  prefix-sid-map
   address-family ipv4
    1.1.1.0/32 10 range 40
    2.1.1.0/24 100 range 60
   !
  !
 !
!
router ospf 1
 segment-routing prefix-sid-map advertise-local
!
```

Node1 advertises the prefix-to-SID mappings in two Extended Prefix LSAs, one per prefix-to-SID mapping range. See Example 8-20 and Example 8-21. The IA-flag of the Extended Prefix Range TLV are unset. The NP-flag (PHP-off) and M-flag (Mapping Server) of the Prefix-SID sub-TLV are both set. IETF draft-ietf-ospf-segment-routing-extensions-08 specifies to ignore the NP-flag if the M-flag is set.

Example 8-20: Extended Prefix Range in Area 1, 2.1.1.0/24

```
RP/0/0/CPU0:xrvr-1#show ospf 0 1 database opaque-area 2.1.1.0/24 self-originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 1)

  LS age: 872
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.2
  Opaque Type: 7
  Opaque ID: 2
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000001
  Checksum: 0xffd5
  Length: 48

    Extended Prefix Range TLV: Length: 24
      AF        : 0
      Prefix    : 2.1.1.0/24
      Range Size: 60
      Flags     : 0x0             !! IA:0

      SID sub-TLV: Length: 8
        Flags     : 0x60          !! (NA); NP:1; M:1; E:0;
V:0; L:0
        MTID      : 0
        Algo      : 0
        SID Index : 100

```

Example 8-21: Extended Prefix Range in Area 1, 1.1.1.0/32

```
RP/0/0/CPU0:xrvr-1#show ospf 0 1 database opaque-area
1.1.1.0/32 self-originate


            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area
1)

  LS age: 881
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.3
  Opaque Type: 7
  Opaque ID: 3
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000001
  Checksum: 0xbf7c
  Length: 48

    Extended Prefix Range TLV: Length: 24
      AF        : 0
      Prefix    : 1.1.1.0/32
      Range Size: 40
      Flags     : 0x0             !! IA:0

      SID sub-TLV: Length: 8
        Flags     : 0x60          !! (NA); NP:1; **M:1**; E:0;
V:0; L:0
        MTID      : 0
        Algo      : 0
        SID Index : 10

```

Example 8-22 shows the Active Mapping Policy on Node1. Since there is only a single Mapping Server in this example, the Active Mapping Policy matches the Local Mapping Policy.

Example 8-22: Active Mapping Policy on Node1

```
RP/0/0/CPU0:xrvr-1#show ospf segment-routing prefix-sid-map
active-policy

        SRMS active policy for Process ID 1

Prefix                    SID Index    Range        Flags
2.1.1.0/24                100          60
1.1.1.0/32                10           40

Number of mapping entries: 2

RP/0/0/CPU0:xrvr-1#show ospf segment-routing prefix-sid-map
active-policy detail

        SRMS active policy for Process ID 1

Prefix
2.1.1.0/24
    Source:           Local
    Router ID:        1.1.1.1
    Area ID:          Not set
    SID Index:        100
    Range:            60
    Last Prefix:      2.1.60.0/24
    Last SID Index:   159
    Flags:
1.1.1.0/32
    Source:           Local
    Router ID:      1.1.1.1
    Area ID:        Not set
    SID Index:      10
    Range:          40
    Last Prefix:    1.1.1.39/32
    Last SID Index: 49
    Flags:

Number of mapping entries: 2
```

ABR Node3 propagates both prefix-to-SID mapping ranges from Area 1 to Area 0. Only one is shown in Example 8-23, the other mapping range is equivalent. The IA-flag of the Extended Prefix Range TLV is set since this TLV has been propagated between areas; it is an inter-area mapping in Area 0. The NP-flag (PHP-off) and M-flag (Mapping Server) of the Prefix-SID sub-TLV are both set.

Example 8-23: Extended Prefix Range Area 0, 1.1.1.0/32

```
RP/0/0/CPU0:xrvr-3#show ospf 1 0 database opaque-area 1.1.1.0/32 self-originate


            OSPF Router with ID (1.0.1.3) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 0)

  LS age: 1068
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.6
  Opaque Type: 7
  Opaque ID: 6
  Advertising Router: 1.0.1.3
  LS Seq Number: 80000001
  Checksum: 0x2394
  Length: 48

    Extended Prefix Range TLV: Length: 24
```

```
       AF        : 0
       Prefix    : 1.1.1.0/32
       Range Size: 40
       Flags     : 0x80              !! **IA:1**

       SID sub-TLV: Length: 8
         Flags     : 0x60           !! (NA); NP:1; **M:1**; E:0;
V:0; L:0
         MTID      : 0
         Algo      : 0
         SID Index : 10

```

Example 8-24 shows the Active Mapping Policy on Node4. Note that the router-id of each mapping range is now the router-id of ABR Node3 (1.0.1.3). It is the router-id of the advertising node.

Example 8-24: Active Mapping Policy on Node4

```
RP/0/0/CPU0:xrvr-4#show ospf segment-routing prefix-sid-map
active-policy

        SRMS active policy for Process ID 1

Prefix                    SID Index    Range        Flags
2.1.1.0/24                100          60
1.1.1.0/32                10           40

Number of mapping entries: 2

RP/0/0/CPU0:xrvr-4#show ospf segment-routing prefix-sid-map
active-policy deta$

        SRMS active policy for Process ID 1

Prefix
2.1.1.0/24
    Source:             Remote
    Router ID:          1.0.1.3
    Area ID:            0
    SID Index:          100
    Range:              60
    Last Prefix:    2.1.60.0/24
    Last SID Index: 159
    Flags:
1.1.1.0/32
    Source:         Remote
    Router ID:      1.0.1.3
    Area ID:        0
    SID Index:      10
    Range:          40
    Last Prefix:    1.1.1.39/32
    Last SID Index: 49
    Flags:

Number of mapping entries: 2
```

Node5 then further propagates the prefix-to-SID mappings to Area 2. See Example 8-25. The IA-flag of the Extended Prefix Range TLV is set since this TLV has been propagated between areas; it is an inter-area mapping. The NP-flag (PHP-off) and M-flag (Mapping Server) of the Prefix-SID sub-TLV are both set.

Example 8-25: Extended Prefix Range Area 2, 1.1.1.0/32

```
RP/0/0/CPU0:xrvr-5#show ospf 1 2 database opaque-area 1.1.1.0/32 self-originate


            OSPF Router with ID (1.0.1.5) (Process ID 1)

                Type-10 Opaque Link Area Link States (Area 2)

  LS age: 1254
  Options: (No TOS-capability, DC)
  LS Type: Opaque Area Link
  Link State ID: 7.0.0.9
  Opaque Type: 7
  Opaque ID: 9
  Advertising Router: 1.0.1.5
  LS Seq Number: 80000001
  Checksum: 0xf8b9
  Length: 48

    Extended Prefix Range TLV: Length: 24
      AF        : 0
      Prefix    : 1.1.1.0/32
      Range Size: 40
      Flags     : 0x80             !! IA:1

      SID sub-TLV: Length: 8
        Flags     : 0x60           !! (NA); NP:1; M:1; E:0;
V:0; L:0
        MTID      : 0
        Algo      : 0
        SID Index : 10

```

Example 8-26 shows the Active Mapping Policy on Node7. Here the router-id of each mapping range is the router-id of ABR Node5 (1.0.1.5).

Example 8-26: Active Mapping Policy on Node7

```
RP/0/0/CPU0:xrvr-7#show ospf segment-routing prefix-sid-map
active-policy

        SRMS active policy for Process ID 1

Prefix                    SID Index    Range        Flags
2.1.1.0/24                100          60
1.1.1.0/32                10           40

Number of mapping entries: 2

RP/0/0/CPU0:xrvr-7#show ospf segment-routing prefix-sid-map
active-policy deta$

        SRMS active policy for Process ID 1

Prefix
2.1.1.0/24
    Source:            Remote
    Router ID:         1.0.1.5
    Area ID:           2
    SID Index:      100
    Range:          60
    Last Prefix:    2.1.60.0/24
    Last SID Index: 159
    Flags:
1.1.1.0/32
    Source:         Remote
    Router ID:      1.0.1.5
    Area ID:        2
    SID Index:      10
    Range:          40
    Last Prefix:    1.1.1.39/32
    Last SID Index: 49
    Flags:

Number of mapping entries: 2
```

8.11 Summary

- The Local Mapping Policy contains the prefix-to-SID mappings that are locally configured on a Mapping Server.
- The local Mapping Policy contains a set of valid, non-overlapping prefix-to-SID mappings
- A Mapping Client applies selection rules to all received prefix-to-SID mappings to derive a valid, non-overlapping non-conflicting set of prefix-to-SID mappings.
- Each SR node is by default a Mapping Client.
- OSPF propagates Mapping Server advertisements between areas.

8.12 References

- [draft-ietf-isis-segment-routing-extensions] Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions (work in progress), June 2016, https://datatracker.ietf.org/doc/draft-ietf-isis-segment-routing-extensions.
- [draft-ietf-ospf-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions (work in progress), July 2016, https://datatracker.ietf.org/doc/draft-ietf-ospf-segment-routing-extensions.
- [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://datatracker.ietf.org/doc/rfc4915.
- [RFC7794] Ginsberg, L., Ed., Decraene, B., Previdi, S., Xu, X., and U. Chunduri, "IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability", RFC 7794, DOI 10.17487/RFC7794, March 2016, https://datatracker.ietf.org/doc/rfc7794.

[1] Cisco Network Services Orchestrator (NSO), enabled by Tail-f, http://www.cisco.com/go/nso

[2] If, due to a reconfiguration, a mapping range is moved from one LSP to another, then it is possible that a receiving router receives two LSPs

containing conflicting mapping ranges from the same advertising router. Of course this will be for a short period of time until the advertising router advertises a new version of the second LSP without the conflicting entry.

[3] [draft-ietf-isis-segment-routing-extensions]

9 TOPOLOGY INDEPENDENT LFA (TI-LFA)

9.1 Introduction

Fast reroute is a mechanism to reduce traffic restoration time when a failure occurs in the network. Most often, the considered failure is a link, but it could also be a node or a shared risk link group (SRLG – see reminder box later in this section).

The node providing protection is called the point of local repair (PLR). The PLR is directly connected to the link. It pre-computes the fast reroute (FRR) repair path in advance of the failure and installs the solution in the data plane. Upon detecting the failure of its link, the PLR enables the precomputed FRR solution and triggers the IGP convergence.

A FRR solution is expected to restore the traffic within 50 msec of the failure occurrence with the link detection time usually estimated as 10 msec. This is the case for SONET/SDH links or for bidirectional failure detection (BFD) monitored links (BFD interval 3 msec, 3 consecutive no-show indicate the failure). If the link detection time is set to be longer (e.g. for dampening link flaps) then the time to restore traffic will increase correspondingly.

The activation of the precomputed FRR solution can either be prefix-dependent or prefix-independent. A true FRR solution requires a prefix-independent solution – i.e. the number of prefixes, whose primary paths are impacted by the failure and which need to get their backup paths activated, does not have any impact on the time to activate the precomputed FRR solution. A prefix-independent solution typically requires a hierarchical FIB data plane. The activation of the precomputed

FRR solution, assuming a prefix-independent implementation, typically takes ~20 msec.

REMINDER: Flattened and hierarchical FIB

Traditionally the structure of the Forwarding Information Base (FIB) was flat. When programming the Routing Information Base (RIB) entries in FIB, each entry was fully resolved and a pointer to the outgoing adjacency was added to the entry. The adjacency contains the outgoing interface and Layer-2 rewrite information. Also the FIB entries of the BGP entries were flattened. The FIB would look like the one shown in Figure 9-1. B1, B2, and Bn are BGP prefixes. I1, I2, and I3 are IGP prefixes. In the illustration each prefix entry directly links to adjacency Adj1.

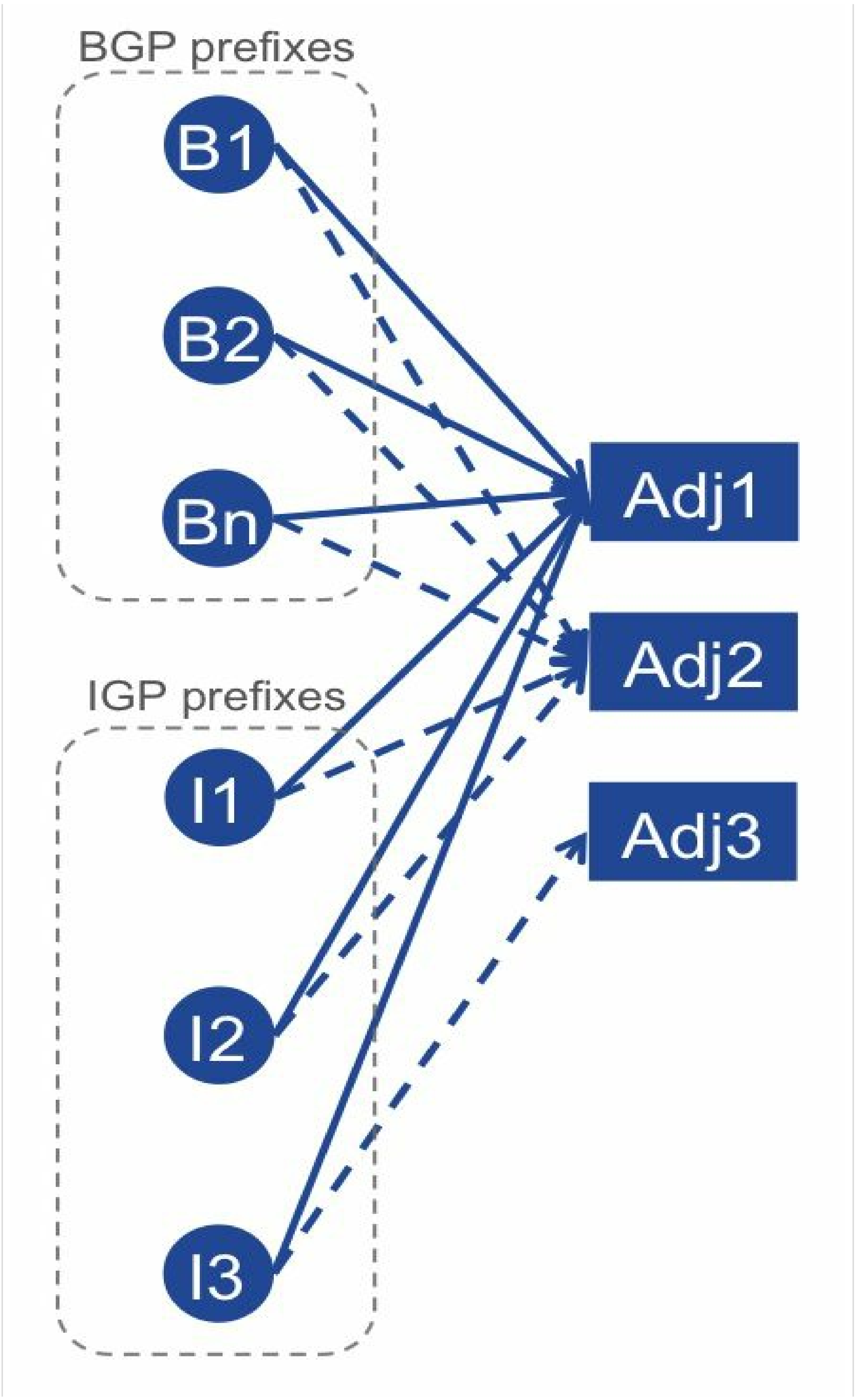

BGP prefixes
B1
B2
Bn
IGP prefixes
I1
I2
I3
Adj1
Adj2
Adj3

Figure 9-1: Flattened FIB structure

To activate IPFRR with such structure, it would require walking over each entry to activate the repair path. This solution would be prefix-dependent; the protection for each prefix would be activated sequentially.

A hierarchical FIB does not flatten the FIB, but uses pointers for the dependencies between FIB elements. Several levels of indirection are introduced in the structure. See the illustration of such FIB in Figure 9-2. Each BGP prefix entry has a pointer to an element called BGP Path list. This BGP Path list contains one or more nexthops for the BGP prefixes. If BGP Multipath is used then the BGP Path list contains more than one nexthop and traffic is then load-balanced over all the nexthops in the BGP Path list. BGP Parth lists are shared among prefixes; BGP prefixes with the same nexthop(s) point to the same BGP Path list. An entry in the BGP Path list points to an IGP prefix entry. This IGP prefix entry points to an IGP Path list. This IGP Path list contains one or more nexthops. If the destination is reachable via multiple equal cost paths, then the IGP Path list contains an entry for each path. Each entry in the IGP Path list points to the corresponding adjacency. The entry of the path list that is used to forward a packet is selected when forwarding the packet.

Aside of the ECMP, an IGP Path list contains information of the backup path. At the time of failure, the backup path is activated. This is illustrated in Figure 9-2. When Adj1 fails, the backup entries in the IGP Path lists are activated. In the illustration, prefix I1 and I2 would use backup path via Adj2, and prefix I3 would go via Adj3.

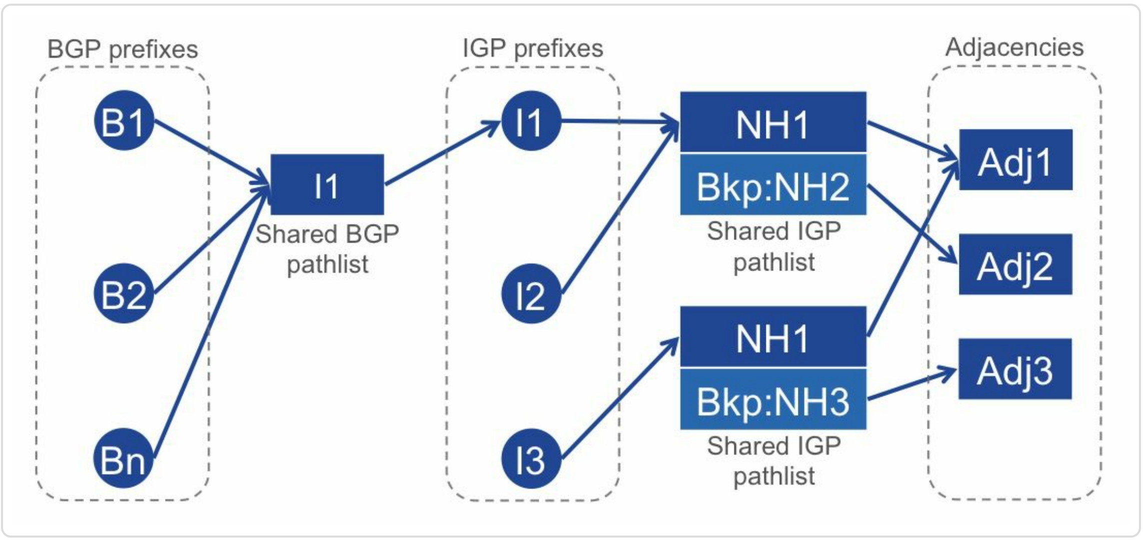


Figure 9-2: Hierarchical FIB structure

The IGP convergence primarily depends on the number of prefixes in the IGP. A modern IGP implementation should converge 1000 prefixes within a few 100's of msec after the failure occurrence. The chart in Figure 9-3 illustrates this behavior. After a failure, each node starts its convergence process. After computing the new Shortest Path Tree (SPT), a node updates the forwarding entries (prefixes) sequentially[1]. The Y-axis shows the loss of connectivity time for a prefix. This is the period of time between the occurrence of the failure and the time that the connectivity for the prefix was restored. For the graph labeled "IGP", this is the time that the prefix entry was updated in the forwarding table. The X-axis indicates the position of each prefix in the order that they are updated. For example the first prefix is updated 100 ms after the failure. The last prefix (the 5000th prefix in this example) is updated 1100 ms after the failure. The 1000th prefix was updated 300 ms after the failure. The point of the FRR solution is to repair the traffic loss until the IGP has converged and the "post-convergence" paths are installed in the data plane. IPFRR restores connectivity for all prefixes at the same time, the time that IPFRR was activated. In Figure 9-3 this happens 20 ms after the failure, see graph labeled "IPFRR". Note that both IGP and IPFRR occur in parallel: while IPFRR is active, IGP computes and updates the new forwarding entries.

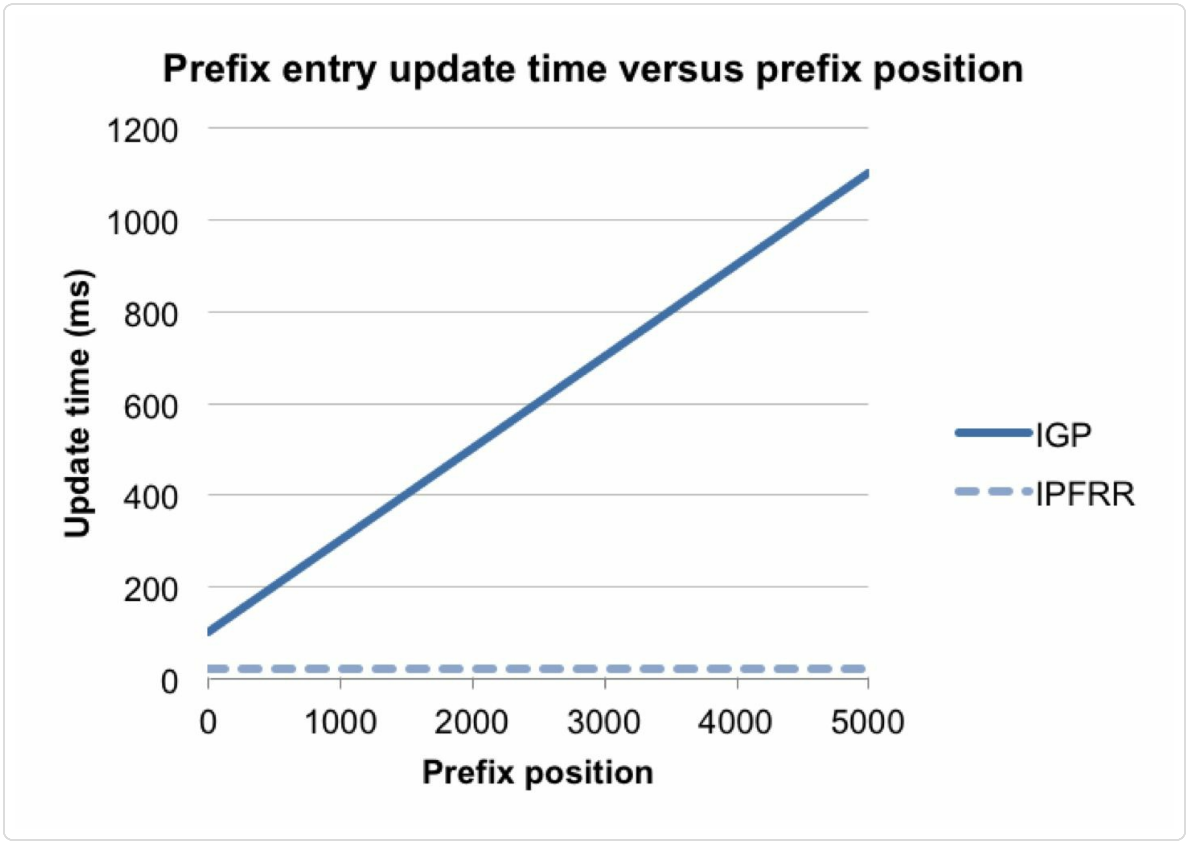


Figure 9-3: Prefix update time versus prefix position

There are two types of FRR solutions: facility-based and prefix-based.

A facility-based FRR solution computes a single backup path from the PLR, around the facility (e.g. link) and back to the next-hop of the PLR over the protected facility. Upon loss of the link, all the impacted traffic is rerouted via this single backup path and is steered to the other side of the link, from where the traffic splits into a set of flows, each travelling to its own destination. See the illustration in Figure 9-4. The facility backup-path (labeled "Facility backup") steers the traffic to the other end of the protected link between Node2 and Node3, from where the traffic flows to the destinations. The traffic streams to destinations Node6 and Node9 share the same facility backup path.

RSVP-TE FRR and SONET/SDH are examples of facility-based FRR solution.

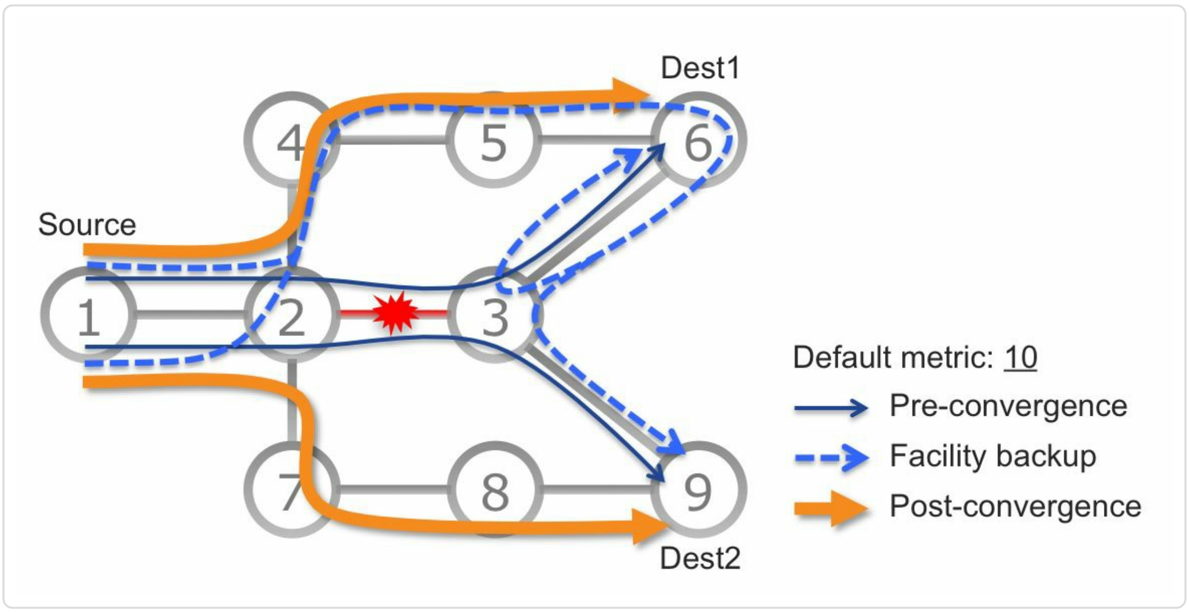


Figure 9-4: Facility backup illustration

The main merit of these solutions is that they were historically the first ones to be implemented.

They have significant drawbacks when applied to IP networks: they steer all the traffic along the same backup path (not using ECMP) and all the way to the other side of the protected link (non optimum).

IPFRR is a prefix-based FRR solution; the PLR pre-computes an individual backup path on a per destination/prefix basis.

This is significantly better for IP networks for the following reasons:

- ECMP: each backup path could be different and hence, upon activation of the FRR solution, the impacted traffic is split into flows and each flow is steered along its optimum backup paths. The impacted traffic is not concentrated along the same backup path.

- Optimality: in IP networks, the optimum backup path most often avoids the other side of the failed link. A facility-based FRR solution is largely sub-optimal when applied to IP networks: it forces the traffic all the way to the other side of the failed link for no reason. In fact, a facility-based FRR solution does this because it was designed for a circuit paradigm and hence the protected traffic had to be rerouted to the other side of the link to stitch back into the circuit state. In IP networks, IP packets have all the routing information they need: i.e. the destination address.

MPLS/LDP traffic also benefits from IPFRR by leveraging the backup paths computed by IPFRR.

> "Stefano Previdi and I initiated the IPFRR research in the Brussels Cisco cafeteria in 2001. It was the early days of MPLS. I was focusing on the FRR/TE/QoS aspect and Stefano on VPN. It was very clear that RSVP-TE FRR was suffering from its circuit's nature and was suboptimal for IP. We wanted to find an IP-based solution that would be optimal for IP (from an ECMP, capacity and routing viewpoint). We called the project IPFRR to stress the centrality of IP in the solution."
>
> — *Clarence Filsfils*

Node2 in Figure 9-5 has protection enabled for the link between Node2 and Node3; this link is the protected component. The primary path from Node2 to destinations Node6 and Node9 traverse the protected component, as indicated by the arrows labeled "Pre-convergence". Node2 computes a repair path for each of these destinations, as indicated by the arrows labeled "IPFRR". Upon failure of the protected component, Node2 deviates the traffic towards these destinations via their repair paths. While the traffic is reaching its destination via the repair paths, IGP converges and installs new forwarding entries. The traffic is no longer forwarded via the repair path, but via the new paths, as indicated by the arrows labeled "Post-convergence".

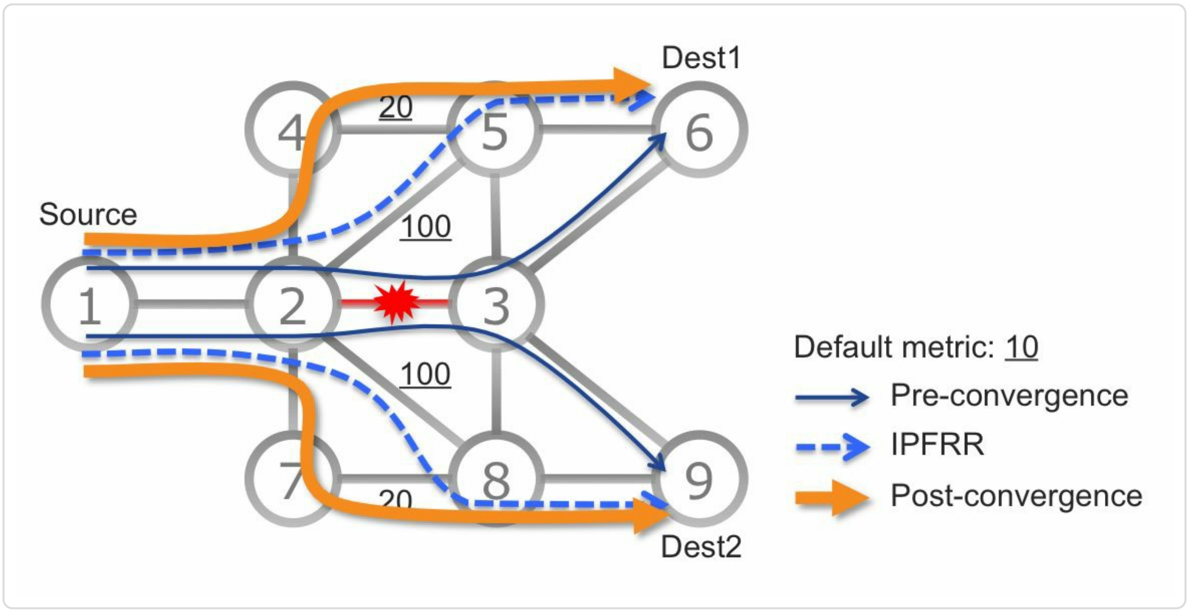


Figure 9-5: Local protection illustration

Fast reroute provides local protection for traffic traversing the protected component. It is local since the node that is directly upstream of the failed component activates the mechanism. It is the first node to detect the failure of the locally connected component, hence eliminating any delays to propagate the failure condition through the network. The other nodes in the network do not have to be aware of the protection activation and do not do anything.

HIGHLIGHT

A FRR solution hides the loss of connectivity that starts when a link, node or SRLG fails and lasts until the IGP installs the post-convergence paths of the impacted destinations (the destinations which were initially routed via the failed link, node or SRLG).

A FRR solution is computed by a node directly connected to the link, node or SRLG whose failure needs to be protected. It is called the point of local repair (PLR).

The PLR pre-computes the FRR repair path prior to the occurrence of the failure.

We seek a per-destination FRR repair path: the backup (repair) path is computed and optimized on a per-destination basis.

We seek a prefix-independent FRR solution: the time to activate the pre-computed FRR repair path does not depend on the number of destination entries that need their repair path activated.

A FRR solution only depends on intelligence of the PLR. The other nodes in the network do not need to do anything.

We seek a 50-msec FRR solution. We assume that the time to detect the local failure is sub-10 msec. The destination-independent nature of the FRR implementation and the pre-computation of the backup/repair paths ensure that the activation is < 40 msec.

We seek an FRR solution optimized for IP, not for circuits.

The IPFRR solution family is called as such to stress its "IP" nature and its optimization for IP networks.

MPLS/LDP traffic benefits from an IPFRR.

The IPFRR research first produced loop-free alternate (LFA). The LFA coverage was then extended with remote LFA (RLFA). Finally, thanks to SR, the research was completed with the delivery of topology-independent LFA (TI-LFA), a complete and optimum IPFRR solution.

A fast-reroute mechanism must pre-compute the repair path to protect for a particular failure. This failure can be a link failure, a node failure or a shared-risk link group (SRLG) failure. The failure type to protect for is usually configured as part of fast-reroute configuration.

Figure 9-6 illustrates the link, node and SRLG protection types. Node5 in the topology applies the different protection models to protect traffic to destination Node3.

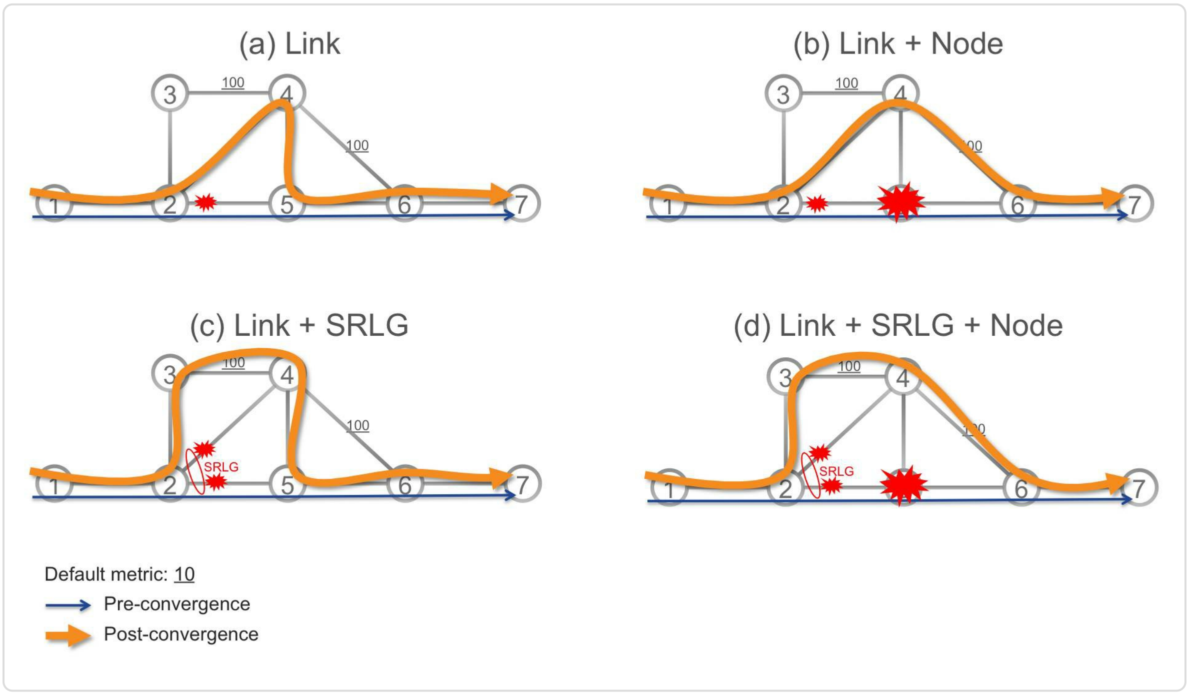


Figure 9-6: Link, node, and SRLG protection

Link protection means that the repair path avoids the outgoing link along the primary path to the destination. See Figure 9-6 (a).

Node protection means that the repair path avoids the first-hop node along the primary path to the destination. Node protection implies link protection. See Figure 9-6 (b).

SRLG protection means that the repair path avoids all the links that share an SRLG with the primary path to the destination. The Cisco IOS XR implementation protects against failure of local interfaces that belong to the same SRLG as the primary path, the so-called *local* SRLG protection. SRLG protection implies link protection. See Figure 9-6 (c).

REMINDER: Shared Risk Link Group (SRLG)

A shared risk link group (SRLG) is a group of network links that share a common (physical) resource (such as a cable, conduit, node or substructure). A failure of that resource will cause the failure of all links of the group. For example, two fibers in the same conduit or multiple wavelengths sharing the same fiber would be in the same SRLG. A link may belong to multiple SRLG's.

A number identifies each SRLG, that number is attached to each link that is member of that SRLG. IETF RFC 4202 states: "An SRLG is identified by a 32 bit number that is unique within an IGP domain. The SRLG Information is an unordered list of SRLG's that the link belongs to."

SRLG identifiers are advertised in IGP. The SRLG number is advertised in a SRLG TLV in ISIS (IETF RFC 4205). It is advertised in a SRLG sub-TLV of the Link TLV in the OSPF TE Opaque LSA (IETF RFC 4203).

Node and SRLG protection can be combined, and imply link protection as well. See Figure 9-6 (d).

The Cisco IOS XR implementation supports:

- Link, node or local SRLG protection with LFA.
- Link protection with remote LFA.
- Link, node or local SRLG protection with TI-LFA.

9.2 Loop-Free Alternate

The simplest way to come up with a repair path for a destination D is for the PLR to find a directly connected neighbor whose shortest path to D does not traverse the protected component. Such neighbor is called a loop free alternate (LFA).

Upon failure of the protected component, the PLR reroutes the traffic destined to D via the pre-computed LFA. The LFA neighbor receives this traffic and naturally (i.e. unaware of the failure) forwards it to the destination along its shortest-path. This is important because we want to restore traffic within such a short time (50 ms) where we cannot expect for the LFA neighbor to have completed its IGP convergence. It must be ensured that the LFA neighbor that we pick would forward traffic towards the destination using both its old or new forwarding state after the failure event.

The LFA neighbor selected by the PLR is called a "release point" for destination D: a packet released at the LFA naturally gets to the destination D without crossing the protected component or coming back to the PLR.

A PLR may not have any LFA for a particular destination, if the topology does not allow it. In this case, no protection will be installed for this destination and traffic would get dropped until IGP convergence.

"Classic" loop-free alternate is specified in IETF RFC 5286. The prefix "classic" is added to distinguish it from other types of LFA in this book. We recommend reading IETF RFC 6571 ("LFA Applicability in Service

Provider (SP) Networks”) to learn use-cases and design guidelines for LFA.

From inspection of the topology in Figure 9-5, one can see that Node2 uses Node4 as an LFA for destination Node6. The shortest path from Node4 to Node6 goes via Node5 and does not traverse the protected component, the link between Node2 and Node3. Similarly, Node2 uses Node7 as an LFA destination Node9. Note that Node2 computes per-destination backup paths: Node4 for destination Node6 and Node7 for destination Node9.

Note the optimality of having a backup path per destination. A circuit-based solution such as RSVP-TE FRR would have used a single backup path (e.g. 2-7-8-3) and would have steered all the repaired traffic to the other end of the failed link (Node3). This is largely suboptimal from a capacity and ECMP viewpoint but as well from a routing and latency viewpoint: the repaired traffic to Node6 surely should not go via Node7 and Node8. The repaired traffic to Node9 surely should not go back via Node3.

The formalization of the LFA idea is simple (Equation 9-1)

Equation 9-1: LFA Basic Loop-free condition

$$\text{Dist}(N, D) < \text{Dist}(N, PLR) + \text{Dist}(PLR, D)$$

- Dist(A, B) is the shortest distance from A to B
- N is a neighbor of the PLR
- D is the destination under consideration

The equation simply formalizes that, for a PLR, its neighbor N is a LFA for destination D if and only if the shortest-path from N to D does not come back via the PLR. In other words: if PLR sends a packet with destination D to N then N will not loop it back to the PLR if the condition is satisfied. It is the loop-free criterion.

Now evaluate the condition of Equation 9-1 in the network topology of Figure 9-5 with PLR=Node2 and D=Node6. When first evaluating N=Node7, the condition becomes:
Dist(N, D) < Dist(N, PLR) + Dist(PLR, D)
→ Dist(Node7, Node6) < Dist(Node7, Node2) + Dist(Node2, Node6)
→ 30 < 10 + 20
→ False → Node7 is not a LFA for destination Node9.

The shortest path from Node7 to Node6 indeed traverses the link from Node2 to Node3.

When evaluating N=Node4, the condition becomes:
Dist(N, D) < Dist(N, PLR) + Dist(PLR, D)
→ Dist(Node4, Node6) < Dist(Node4, Node2) + Dist(Node2, Node6)
→ 20 < 10 + 20
→ True → Node4 is a LFA for destination Node6.

Classic LFA is simple and very often possible. IETF RFC 6571 analyzes many real data sets and shows that LFA is possible for 89% of the destinations in a typical backbone. IETF RFC 6571 also provides design guidelines to ensure 100% coverage.

Coverage means the percentage of pairs (PLR P, destination D) such that the PLR P has an LFA for the destination D.

Classic LFA is a simple protection mechanism, but it has limitations and disadvantages:

- It does not provide protection in all cases, its coverage is topology-dependent
- It does not always provide the most optimal backup path

A suitable (classic) LFA does not always exist; it depends on the network topology, metrics, and the component to be protected. Although classic LFA provides protection for the majority of destination prefixes in most networks, it cannot provide protection for all destinations in all topologies. It is “topology dependent”. It is also said that classic LFA does not provide 100% protection coverage.

In the example network topology in Figure 9-7, Node2 wants to protect traffic going to destinations Node8 and Node5 against failure of the link to Node3. From inspection of the network topology one can see that Node6 is an LFA for destination Node8 on Node2. Using Equation 9-1, one can mathematically verify that Node6 is an LFA for destination Node8 since it fulfills the basic loop-free condition:
Dist(N, D) < Dist(N, PLR) + Dist(PLR, D), with D=Node8, PLR=Node2, and N=Node6
→ Dist(Node6, Node8) (=10) < Dist(Node6, Node2) (=10) + Dist(Node2, Node8) (=20)
→ 10 < 10 + 20
→ True → Node6 is an LFA for destination Node8.

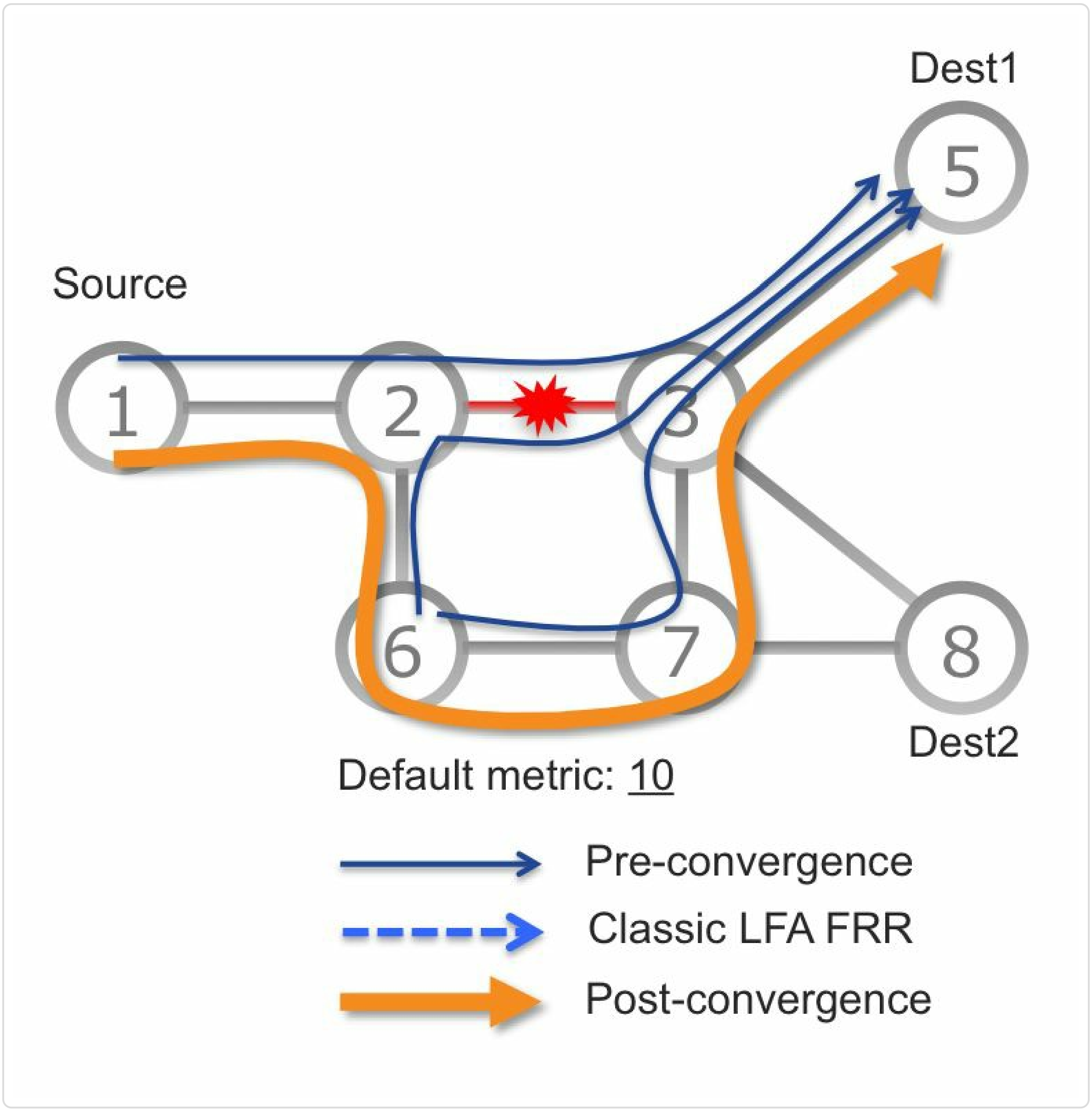


Figure 9-7: Classic LFA is topology dependent

However, Node2 does not have a classic LFA for destination Node5. Node2 has only one alternative neighbor: Node6. However the shortest path from Node6 to destination Node5 is an ECMP with one of the paths crossing the link between Node2 and Node3. The other equal cost path from Node6 to Node5 goes via Node7. Mathematically, Node6 does not fulfill the LFA Basic Loop-free condition: Dist(Node6, Node5) (=30) is not smaller than Dist(Node6, Node2) (=10) + Dist(Node2, Node5) (=20).

Therefore Node6 is not an LFA for destination Node5. Note that a backup path from Node2 to Node5 exists: Node2→Node6→Node7→Node3→Node5. This is also the path the traffic will follow after convergence completes, the post-convergence path. Since it is not loop-free (Node6 is not an LFA as concluded above) this backup path cannot be used with the classic LFA functionality. Half of the traffic would loop between Node6 and Node2.

The other disadvantage of classic LFA: it does not always provide the most optimal backup path. If a classic LFA is found, it provides protection, but possibly not via the most desired repair path.

In the example network topology in Figure 9-8, Node2 wants to protect traffic going to destinations Node8 and Node5 against failure of the link to Node3. From inspection of the network topology one can derive that Node4 is an LFA for the destination Node5 since the shortest path from Node4 to Node5 does not traverse the protected link. Node4 also mathematically fulfills the LFA loop-free condition:
Dist(N,D) < Dist(N, PLR) + Dist(PLR, D), with PLR=Node2, D=Node5, and N=Node4
→ D(Node4, Node5) (=110) < D(Node4, Node2) (=100) + D(Node2, Node8) (=20)
→ 110 < 100 + 20
→ True → Node4 is an LFA for destination Node5.

The other neighbor of Node2, Node6, is not an LFA for destination Node5. This was concluded in the previous example for the network topology in Figure 9-7, and is also true for the topology in Figure 9-8.

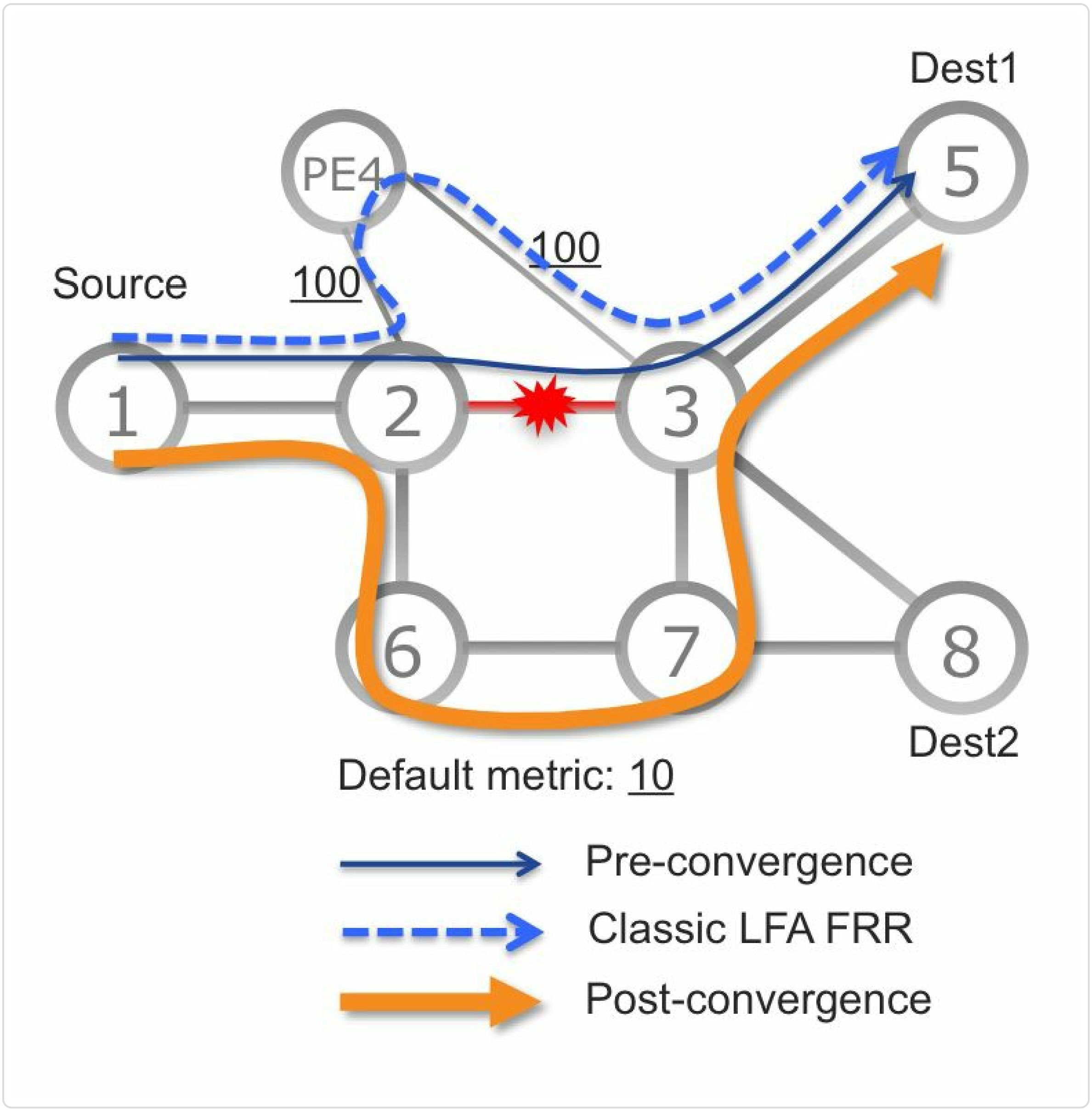


Figure 9-8: Classic LFA may provide sub-optimal repair path

Note that the only available LFA for destination Node5, Node4, is a Provider Edge node. The operator has configured high metrics on the links connecting Node4 to the core. The operator clearly does not want Node4 to be used for transit traffic, which is a common planning rule. The high metric links typically have low capacity, which may lead to congestion and packet loss if used as repair path. Still Classic LFA selects Node4 as an LFA since in this case it is the only available LFA.

The IETF RFC 7916 describes other cases where the classic LFA mechanism does not pick the desired repair path. Sometimes the desired result may be achieved with operational intervention. It may be possible to manually tweak the so-called tiebreakers to realize the desired repair path. The tiebreakers are the criteria that are used to select one LFA from a number of possible LFAs, see IETF RFC 5286. However, this case-by-case configuration is not always possible and it anyway increases the operational complexity of the classic LFA solution.

When analyzing the most desired repair paths in a network, it turns out that the best repair path is always the post-convergence path, the path that the traffic will follow after IGP has converged. This is exactly the repair path that Topology-Independent LFA provides automatically.

Going back to the example in Figure 9-8. Note that an optimal backup path from Node2 to Node5 exists: Node2→Node6→Node7→Node3→Node5. This path does not cross any undesired nodes or links. It is also the path the traffic will follow after convergence completes, the post-convergence path.

HIGHLIGHT

Classic LFA is simple and often available.

IETF RFC 6571 analyzes many real data sets and shows that LFA is possible for 89% of the destinations in a typical backbone.

IETF RFC 6571 provides design guidelines to ensure 100% coverage.

TI-LFA improves classic LFA in two ways: it guarantees 100% coverage in any topology and it ensures the optimality of the backup path.

9.3 Remote LFA

The previous section highlighted that an LFA cannot always be found. There is not always a directly connected neighbor that is able to steer packets to their destination without traversing the protected component.

IETF RFC 7490 specifies an extension to LFA, called Remote LFA (RLFA). RLFA extends the protection coverage of classic LFA by using “virtual LFAs”. These virtual LFAs are remote nodes that can function as an LFA release point.

In case of a component failure, the protecting node (point of local repair or PLR) can direct traffic towards such remote LFA, where the traffic is then released. From there the traffic travels to its destination following the regular shortest path, without looping back or traversing the protected component. RLFA is typically used if no directly connected LFA is available.

In an RLFA solution, the PLR steers the traffic to the RLFA node via a tunnel. In MPLS networks, the tunnel is typically an LDP LSP.

The RLFA release point fulfills the LFA loop-free condition in Equation 9-1.

The PLR computes the list of possible RLFA nodes for a destination D based on the following two conditions, respectively called P and Q:

- **P**: the shortest path from the PLR to the RLFA candidate must not traverse the protected component C (e.g. a link, a node, or a local SRLG). This applies for all the ECMP paths along the shortest path.

- **Q**: the shortest path from the RLFA candidate to the destination D must not traverse the protected component C.

The set of nodes satisfying the first criterion is called the P-space P(PLR, C) of a node PLR with respect to a component C. The P-space P(PLR, C) is the set of nodes such that the shortest-path from the PLR to them does not cross the component C. Nodes in the P-space P(S, C) can be reached from the PLR irrespective of the state of the protected component C. A PLR can compute its P-space for a local component C to protect, by computing its (normal) Shortest Path Tree and pruning all the nodes that are reached via component C. A possible algorithm to compute the P-space is described in IETF RFC 7490.

The PLR has full control of the first hop of the repair path. It can use any of its adjacent nodes as the first hop of the repair path, regardless of the shortest path. Consequently, if a neighbor of the protecting node can reach a destination without traversing the protected component, then the protecting node can also reach that destination without traversing the protected component. So, it can reach all the nodes in all of its neighbors' P-spaces. The union of the P-spaces of PLR's neighbors is called the Extended P-space. Using the Extended P-space may expand the choice of possible RLFA nodes. Pext(PLR, C) is the Extended P-space of a PLR with respect to a component C.

The set of nodes satisfying the second criterion is called the Q-space Q(D, C) of a destination D with respect to component C. Node N is in the Q-space Q(D, C) if N's shortest path to D avoids C. Nodes in the Q-space can reach the destination, regardless the state of the protected component. Note that a classic directly connected LFA that protects destination D against the failure of component C is a node of the Q-space Q(D, C). An

LFA can reach destination D without traversing component C which is exactly the property of nodes in the Q-space.

For link protection, the Q-space of the node at the remote end of the protected link is used in practice. This Q-space is used as a substitute for the Q-space of each destination reached through that next-hop. This approximation largely reduces the complexity of calculating possible RLFAs, at the expense of potentially suboptimal repair paths. To obtain this Q-space, compute the reverse Shortest Path TreeTo obtain the reverse SPT, compute the normal SPT, but use the link metric in the direction from the next-hop towards the root of the tree instead of the link metric in the direction away from the root.] rooted at the remote end of the protected link and prune all the nodes that reach the root of the SPT via the protected link. A possible algorithm to compute the Q-space is described in IETF RFC 7490.

A Remote LFA is a node that satisfies both the P and Q conditions. Therefore an RLFA is commonly known as “PQ-node”. It is at the intersection of the P- and Q-spaces.

If the protected component C fails, the PLR can steer the protected packets to the RLFA without crossing C (property of P-space nodes), and from the RLFA release points, the packets can reach their destination without crossing C (property of Q-space nodes).

If there are multiple RLFA candidates, it is recommended to select the closest from the PLR, as this maximizes the load balancing possibilities for the traffic travelling from RLFA to their destination.

In the network topology of Figure 9-9 Node2 wants to protect destination Node5 against failure of the link to Node3. The primary path to Node5 crosses the protected link. Node2 has no LFA to protect destination Node5, since the only alternate neighbor, Node6, loops (part of) the traffic back to Node2. Therefore Node2 searches for a possible RLFA.

Node2 computes the P-space P(Node2, link2-3): {Node1, Node2, Node6}. From inspection of the topology, one can see that the shortest path from Node2 to these destinations does not traverse the protected link. The Extended P-space in this example is the union of Node2's P-space and Node6's P-space: P(Node2, link2-3) ∪ P(Node6, link2-3) = {Node1, Node2, Node6, Node7, Node8}. Indeed, if Node2 steers packets with destination Node7 towards Node6, then Node6 forwards these packets via the shortest path to Node7 without traversing the protected link. The same is true for Node8.

Node2 then computes the Q-space Q(Node5, link2-3): {Node3, Node5, Node7, Node8}. Indeed, any of these nodes can reach Node5 via the shortest path without traversing the protected link.

The intersection of Extended P-space and Q-space is: {Node7, Node8}. Node7 is closest to Node2, therefore Node2 selects Node7 as RLFA (PQ-node). Upon failure of the link to Node3, Node2 steers the traffic to destination Node5 via Node7. The traffic can reach Node7 (property of P-space) and from Node7 it can reach Node5 (property of Q-space).

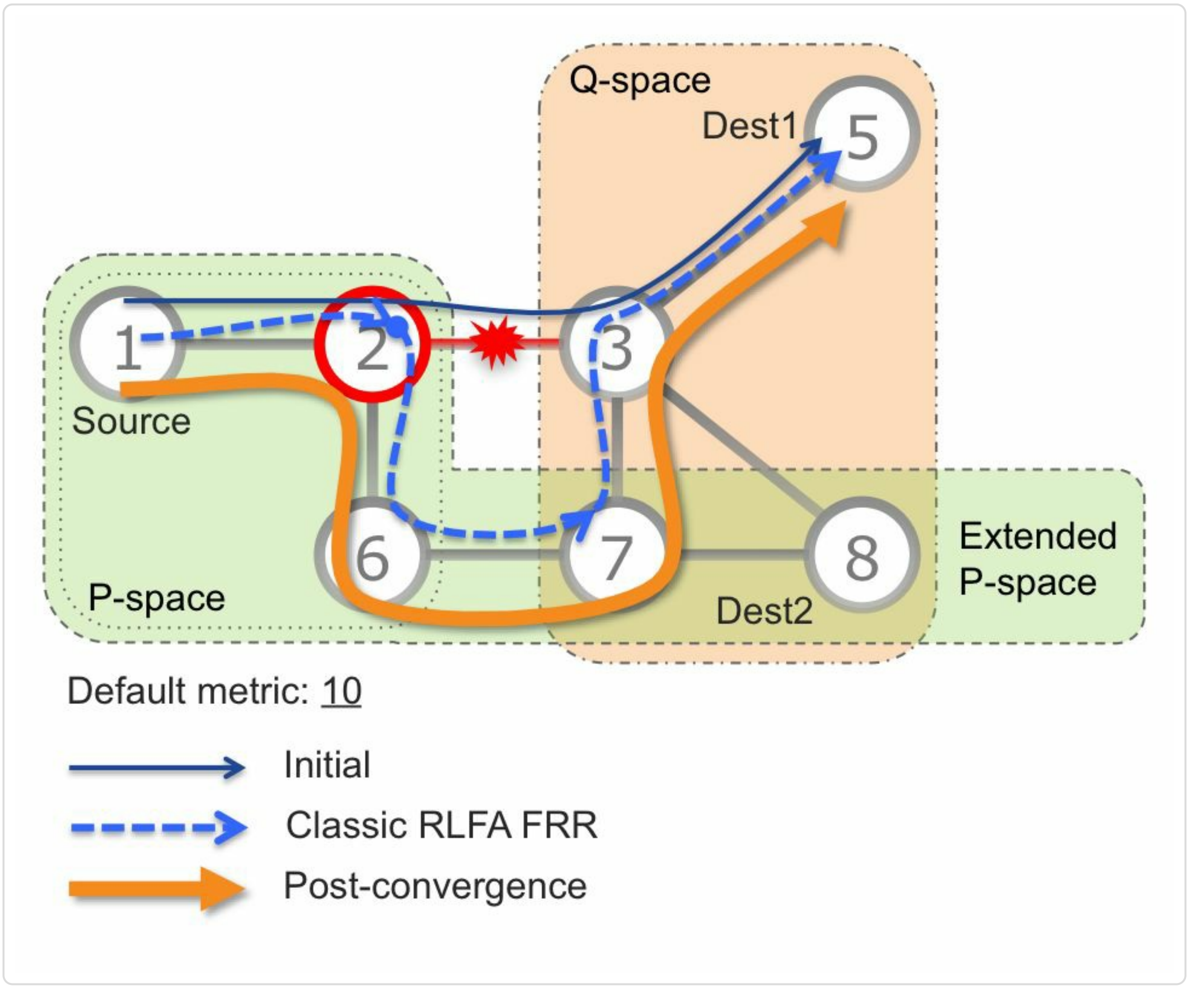


Figure 9-9: RLFA example

To steer the protected traffic via the RLFA to its destination, the protecting node needs to encapsulate or tunnel the packets to the RLFA. Typically the RLFA solution uses LDP as the tunnel transport. In that case the protecting node pushes the LDP outgoing label bound to the FEC of the RLFA node on the packets of the repair path. Then it becomes more complex. If the protected traffic is transported using LDP labels then the protecting node has to know which label the RLFA has allocated for the protected destination. The RLFA forwards packets with that label to the destination. LDP labels are signaled hop-by-hop, only the direct LDP neighbors learn about the local LDP labels. Therefore, for the protecting node to learn the local LDP labels from the RLFA, a (targeted) LDP session must be established between these two nodes. Such (targeted) LDP session is required between each PLR/RLFA pair.

In the network topology of Figure 9-9, Node2 has to establish a targeted LDP session with RLFA Node7. Node2 then learns which local LDP label Node7 has allocated for destination Node5: LDP(Node7, D:Node5). Node2 learns the LDP label it must use to reach Node7 from its downstream neighbor Node6: LDP(Node6, D:Node7). When the link to Node3 fails, Node2 steers the LDP traffic with destination Node5 via Node7. Node2 swaps the top label of the incoming packets with LDP(Node7, D:Node5). This label will bring the packets from Node7 to their destination Node5. On top of that, Node2 also pushes a label to steer the packets to Node7.

Remote LFA extends the protection coverage compared to directly connected LFA, however RLFA does not provide a guaranteed complete coverage. The coverage is still topology dependent. And the selected RLFA may not provide the optimal backup path. And it comes at a cost: RLFA requires targeted LDP sessions between the PLRs and RLFAs.

HIGHLIGHT

RLFA extends classic LFA.

The combined coverage reaches 99% in real data sets analyzed in IETF RFC 7490.

RLFA extends the coverage of the classic LFA solution. IETF RFC 7490 analyses real data sets and shows that LFA+RLFA have a combined 99% coverage of the analyzed backbone networks.

RLFA does not improve the lack of optimality in some cases and does not provide the 100% coverage guarantee.

> “Deployment of LFA could be considered as a quick win in a IP network that does not mandate a guaranteed traffic protection: it is very simple to deploy (usually one CLI command), is very simple to understand, and it has an insignificant scaling impact while providing an almost good coverage in usual topologies. Implementing remote LFA adds an additional level, while it is also simple to deploy (one CLI command), it may be harder to operate as the RLFA candidate would need to accept TLDP sessions from a PLR (additional feature required everywhere in the network). With RLFA it is hard to predict (without a simulation tool) which node would be the best RLFA candidate for a particular destination from a particular PLR. Depending on the design and network size, a particular RLFA candidate may be used by plenty of PLRs, leading to massive TLDP session setup that may add scaling considerations.
>
> LFA and RLFA may not use an optimal repair path as the base tie-breaking rules defined in RFC 5286 and RFC 7490 are really simple. Using a non-optimal path may cause some network troubles such as transient link congestion: imagine if a high bandwidth core link is protected through a path using low bandwidth access links! RFC 7916 describes those considerations and introduces a policy framework where an operator can better drive the choice of the LFA/RLFA candidates.
>
> Even if such policy implementation helps to solve fast reroute traffic steering, it adds an additional level of operational complexity.
>
> My recommendation would be to consider deployment of LFA and RLFA when a guaranteed protection is not mandated and when repair path optimality is not a concern.”
>
> — *Stéphane Litkowski*

9.4 Topology Independent LFA

In the previous sections it became clear that classic LFA has great benefits, but it also has limitations and disadvantages. Classic LFA cannot protect all destinations in all networks; it is topology dependent. A disadvantage of LFA is that even if one or more LFAs exist, the most optimal one is not always provided and manual tweaking is required. This increases the operational complexity.

Remote LFA extends the coverage to 95-99% of the destinations, but it also does not always provide the most desired repair path. On top of that it adds more operational complexity by requiring a targeted LDP session to the RLFAs to protect LDP traffic.

Topology independent LFA (TI-LFA) provides a solution for these limitations while maintaining the simplicity of the IPFRR solution. Since TI-LFA uses Segment Routing for the repair path, SR must be deployed in the network. Note that even a partial deployment of SR may already provide TI-LFA protection. This will be discussed further in this section.

Using topology independent LFA as a protection mechanism has the following benefits:

- TI-LFA provides sub-50msec link, node and SRLG protection with 100% coverage.
- TI-LFA is simple to operate and understand. The functionality is contained within the IGP and no additional protocols or signaling is required.
- The repair path is automatically computed by the IGP, no specific

tuning is required.

- By using the post-convergence path as backup path, it prevents transient congestion and suboptimal routing on the backup path.
- TI-LFA can be incrementally deployed; it is a local functionality.
- TI-LFA also protects LDP and IP traffic in addition to Segment Routing traffic.

The most optimal and natural repair path to follow when a failure occurs is the path that the traffic will eventually follow after the IGP has converged: the post-convergence path. Such path is preferred since:

- it is optimal for capacity planning. During the capacity-planning phase of the network, the capacity of a link is provisioned while taking into consideration that such link will be used when other links fail. Consequently a backup path that is also a post-convergence path is largely deterministic and does not disrupt planned capacities.
- it is simple to operate. There is no need to do case-by-case tweaking to select the best LFA among multiple candidate LFAs. Thus reducing provisioning overhead by eliminating the need to tweak LFA selection.
- there are less traffic transitions. Since the repair path is equal to the post-convergence path, the traffic switches path only once.

HIGHLIGHT

The most optimal and natural repair path to follow when a failure occurs is the path that the traffic will eventually follow after the IGP has converged: the post-convergence path

Before SR, the post-convergence path could not be used since in many cases it is not loop-free. It would loop the traffic back to the protecting node. Traffic must be explicitly steered along the post-convergence path to enforce loop-freeness of the path.

Thanks to the source routing traffic steering capabilities of SR, it is always possible to use the post-convergence as a repair path. It suffices to encode it as a list of segments to make it loop-free! This does not create any additional state on the nodes along the repair path, since the repair path is not signaled, but encoded at the source in the packet header. Thus like LFA/RLFA, TI-LFA is also a local mechanism on the PLR.

With TI-LFA, the repair path to protect a destination D against the failure of a component C is the post-convergence path when C fails: i.e. this is the shortest path to destination D with the component C removed from the topology.

With TI-LFA, the PLR encodes the post-convergence path as a list of segments (i.e. a label stack). Upon activation of the FRR solution, the appropriate label stack gets pushed on the repaired traffic.

Let's remind that the computation of the segment lists (i.e. label stack) is per destination. This ensures the optimality of the solution: each destination is repaired along its own post-convergence path.

Figure 9-10 repeats the network topology of Figure 9-8 to illustrate the optimal and automatic selection of the TI-LFA repair path. In this network, Node2 protects traffic to destination Node5. Figure 9-10 indicates that classic LFA steers the traffic to LFA Node4 after a failure of protected link. As indicated before in the classic LFA section, this path is not optimal since it traverses the Edge node Node4 that is connected via lower capacity links. If the protected link would fail, then the shortest path from Node2 to Node5 would go via the post-convergence path Node2→Node6→Node7→Node3→Node5. Prior to the failure, TI-LFA calculates this post-convergence path and derives the Segment List required to steer packets along the post-convergence path without looping back. In this simple example, TI-LFA will encode a single segment in the header of packets on the repair path: the Prefix Segment of Node7. As shown in the RLFA section, Node7 is the PQ-node for destination Node5. Traffic steered via PQ-node Node7 indeed brings the traffic to destination Node5. And from inspection of the network topology, the traffic will indeed follow the post-convergence path.

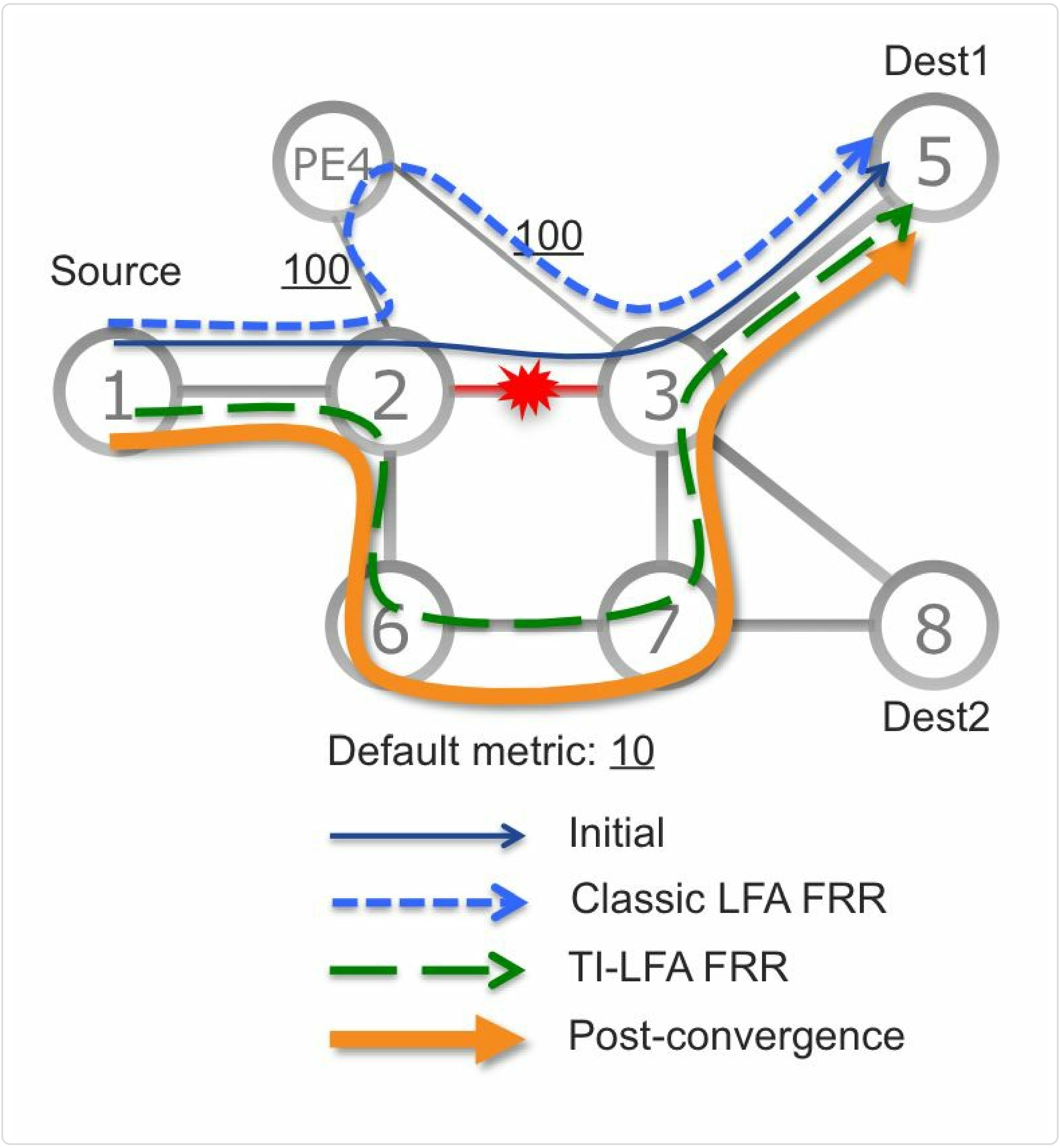


Figure 9-10: TI-LFA uses post-convergence path

HIGHLIGHT: TI-LFA benefits

- Sub-50 msec link, node and SRLG protection
- 100% coverage whatever the topology.
- Simple to operate and understand
- Automatically computed by the IGP, no other protocol required
- No state created outside the protecting state at the PLR; local mechanism

- Optimum: the backup path follows the post-convergence path
- Incrementally deployment
- Besides Segment Routing traffic, it also applies to IP and LDP traffic

"TI-LFA brings a huge benefit over others IPFRR techniques. While LFA and RLFA could not guarantee the protection (even if they provide good coverage), using TI-LFA, an operator knows that traffic protection will always be provided, whatever the design the operator will implement. Before TI-LFA, for an IP or LDP network requiring 100% of traffic protection, an operator had to implement RSVP-TE fast-reroute on each link, which added a lot of complexity (LDP over RSVP tunneling to manage) and still not provided optimal repair path (it was a facility-based FRR solution): traffic was often hairpinning on a lot of links.

TI-LFA now provides a simpler and more natural solution for an IP or LDP network. In addition, even for networks where guaranteed traffic protection is not a concern, TI-LFA is a quick win to deploy (again one CLI command and nothing else to take care of) as soon as your network runs segment routing. Again, deploying segment routing on your network does not mean migrating all your primary traffic to segment routing LSPs. You can easily deploy segment routing, and just use it for repair path built by TI-LFA, so your primary traffic is still pure IP or LDP, it will use segment routing paths only when FRR will be activated, and revert to IP or LDP when IGP will converge.

The traffic optimality of TI-LFA is a huge improvement compared to any other FRR technique. Only RSVP-TE detour FRR was providing such approach in the past but was never deployed because it requires a full mesh of TE tunnels end to end with dedicated FRR path for each tunnel, which does not scale. TI-LFA does not have such scaling concern as there is no state to maintain on intermediate nodes. Traffic optimality is critical especially when bandwidth is rare and expensive: an operator cannot waste it. Moreover having a dual capacity planning policy (one for FRR path, and one for post-convergence path) would increase complexity of the planning and may increase network costs; this does not make sense. So aligning FRR path to post-convergence path removes the need of specific traffic steering policies for FRR and again simplifies the network and its operation."

— Stéphane Litkowski

9.4.1 TI-LFA Computation

This section gives a high-level overview of the steps taken in the TI-LFA repair path computation.

"We spent much research ensuring that the computation of the post-convergence paths and their encoding in SID lists be scalable.

The computation must be done on a per-destination basis every time the topology changes. It must scale to large SP networks that experience frequent changes.

The result of that research has been implemented in Cisco's implementation. The implemented algorithm does not need to be detailed as this is a per-hop behavior."

— Clarence Filsfils

On a per destination basis, the PLR computes the primary path. It detects what is the outgoing link along the primary path. It then computes a shortest-path with this link pruned from the topology. This is the post-convergence path. It then encodes the post-convergence path as a list of segments. This list of segments (i.e. a label stack in SR MPLS) is the precomputed backup path for the considered destination.

The PLR repeats this process for all the destinations in the IGP table.

It is straightforward to apply this process to link, node or SRLG protection. The only modification happens in the pruning step (i.e. what component is removed from the topology to compute the post-convergence path).

Once the primary path is computed to a destination, the PLR looks the TI-LFA policy selected by the operator for the outgoing link along the primary path.

If the policy indicates SRLG, then the PLR prunes the local interfaces that share an SRLG with the primary path. If the policy indicates Node, then the PLR prunes the first-hop node along primary path. Else, the PLR prunes the outgoing link along the primary path.

HIGHLIGHT

TI-LFA offers Link, Node and SRLG protection

9.4.2 Segment List Size Analysis

"It is intuitive that TI-LFA requires few segments to encode the post-convergence path.

First, LFA and RLFA are often (99%) available and require zero or one segment.

Second, the same intuition as the one for SRTE applies. In our daily life, we do not plan our journeys turn by turn; instead we plan them as a very small number of shortest-path hops. Applying this intuition to a real network: one or two intermediate shortest-paths, one or two segments in SR terminology, would be enough to steer the traffic. See chapter 1, "Introduction"."

— Clarence Filsfils

TI-LFA link protection in networks with symmetric metrics (where for each link A-B in the network, the metric of A→B is equal to the metric of B→A) requires maximum two additional segments on the repair path. This is guaranteed and proven (see e.g. reference [FRANCOIS]).

In such networks the P- and Q-spaces overlap or are adjacent for link protection. While the theoretical limit is two additional segments, more than one is rarely needed in reality.

Networks do not always use symmetrical IGP metrics. This metric asymmetry may be intentional, or due to a configuration error. Maybe node or SRLG protection is desired. In these cases, there is no theoretical bound on the number of segments to impose on the backup path. But in reality things are much simpler!

A large Service Provider in France has presented TI-LFA simulation results for their network [MPLSWC14], showing that link protection in their network requires most of the time no additional segment and the very rare worst-case is 2 segments on the backup path.

For node protection, 99.72% of the destinations is protected by using up to 2 segments; with up to 3 segments 99.96% of the destinations is protected; using up to 4 segments provides 100% coverage for node protection.

Analysis of other real network topologies shows similar results.

HIGHLIGHT

Theoretical and real data sets analysis shows that TI-LFA requires shallow segment lists.

Most of the post-convergence backup paths are expressed in 0 or 1 segment.

99.72% of the backup paths analyzed in the real network topology require ≤ 2 segments.

99.96% of the backup paths analyzed in the real network topology require ≤ 3 segments.

No backup path required more than 4 segments

9.4.3 TE Infrastructure for Repair Paths

A link protecting repair path in a network with symmetric metrics requires at most two segments. Also in other circumstances (e.g. if node or SRLG protection is required, or the network has asymmetric metrics) the number of segments needed for the repair path is limited in reality, see section 9.4.2, “Segment List Size Analysis”. However, sometimes more segments (labels in the SR MPLS implementation) are needed on the repair path. The number of segments that can be imposed by a router depends on its hardware platform’s ability. Imposing multiple segments on a packet at a router is not new in traditional MPLS data plane but with Segment Routing and especially some of its Traffic Engineering use-cases, the number of labels to be imposed could be large depending on the position in the network.

To accommodate imposition of more than the platform supported limit for the repair path, the IGPs in Cisco IOS XR use the traffic-engineering (TE) infrastructure. IGP provides the outgoing interface, next-hop and stack of labels of the computed repair path to TE and requests TE to instantiate an SRTE Policy[2] with that information. None of the other TE functionality, such as path validation, is used. With this mechanism the amount of labels on the repair path is increased to a much higher number that is supported for the TE infrastructure.

At the time of writing this book, the maximum number of labels that could be directly imposed were 3 and using TE infrastructure were 10 for platforms like the ASR9000 and NCS6000 series while they could be lesser for certain other platforms or specific line-card models. These numbers are a few orders higher than what is typically required for most TI-LFA deployments and this mechanism is completely abstracted by the implementation so as not to add any complexity or overhead for operator to provision or manage.

To use this functionality the MPLS PIE[3] must be installed on the protecting node where TI-LFA is enabled since the TE infrastructure functionality is contained in that PIE. No TE configuration is required to enable this functionality. Given that in most cases full coverage can be provided with up to three labels on the repair path, this functionality may not be required on a given network.

IGP does not create a different SRTE Policy to protect each individual destination, but shares the same SRTE Policy as a repair path for multiple destinations sharing the repair path.

A limitation of using an automatically instantiated SRTE Policy for the repair path is that all the t repair nodes (i.e. explicit hops) required must be SR-capable, and the nexthop neighbor node on the repair path must be SR-capable. SR/LDP interworking functionality can still be applied on the repair path if intermediate nodes (nodes that are not Segment Endpoint nodes) on the repair path are LDP-only.

The output in Example 9-1 shows an ISIS TI-LFA computed node protecting repair path. Only the SRTE Policy involvement is discussed here. Section 9.5.2.3 discusses the details of this repair path (topology, protection type, etc.). The repair path contains four segments; four labels must be imposed on packets steered on this repair path. For labels stacks larger than three labels, the TE infrastructure is used by IGP to instantiate an SRTE Policy. In this example the SRTE Policy is represented by tunnel-te32783, as displayed in line 6 of the output. The number of this tunnel-te interface is dynamically selected from a default pool starting at 32768. The operator can specify another pool of numbers as illustrated in Example 9-2.

Although no TE configuration is required to use this functionality, at the time of writing this book, it was still required to configure the default IP address of the tunnel-te interface using the configuration `ipv4 unnumbered mpls traffic-eng Loopback0`.

Example 9-1: ISIS TI-LFA node protecting repair path

```
 1| RP/0/0/CPU0:xrvr-1#show isis fast-reroute 1.1.1.6/32 detail
 2|
 3| L2 1.1.1.6/32 [20/115] medium priority
 4|      via 99.1.2.2, GigabitEthernet0/0/0/1, xrvr-2, SRGB
Base: 16000, Weight: 0
 5|          Backup path: TI-LFA (node), via 99.1.5.5,
GigabitEthernet0/0/0/0 xrvr-5, SRGB Base: 16000, Weight: 0
 6|          Backup tunnel: tunnel-te32783
 7|            P node: xrvr-4.00 [1.1.1.4], Label: 16004
 8|            Q node: xrvr-3.00 [1.1.1.3], Label: 30403
 9|            P node: xrvr-8.00 [1.1.1.8], Label: 16008
10|            P node: xrvr-10.00 [1.1.1.10], Label: 16010
11|            Prefix label: 16006
12|        P: No, TM: 110, LC: No, NP: No, D: No, SRLG: No
13|      src xrvr-6.00-00, 1.1.1.6, prefix-SID index 6, R:0 N:1
P:0 E:0 V:0 L:0
```

Example 9-2: SRTE Policy interface number pool

```
mpls traffic-eng
 auto-tunnel p2p
  tunnel-id min 10000 max 19999
```

The details of the SRTE Policy on the protecting node are shown in Example 9-3. Most elements in the output have no significance for the use of the automatically instantiated SRTE Policy of the TI-LFA repair path. The SRTE Policy is `verbatim Segment-Routing`, which means that the IGP dictates how TE constructs the Policy (verbatim = “word-for-word”). The path of the SRTE Policy contains a next-hop (99.1.5.5) and a number of labels {16004, 30403, 16008, 16010}, see lines 37 to 42.

Example 9-3: SRTE Policy instantiated by IGP for TI-LFA repair path

```
 1| RP/0/0/CPU0:xrvr-1#show mpls traffic-eng tunnels 32783
 2|
 3|
 4| Name: tunnel-te32783  Destination: 0.0.0.0  Ifhandle:0x1480
(auto-tunnel for ISIS 1)
 5|   Signalled-Name: auto_xrvr-1_t32783
 6|   Status:
 7|     Admin:    up Oper:   up   Path:  valid   Signalling:
connected
 8|
 9|     path option 10, (verbatim Segment-Routing) type explicit
(_te32783) (Basis for Setup)
10|     G-PID: 0x0800 (derived from egress interface properties)
11|     Bandwidth Requested: 0 kbps  CT0
12|     Creation Time: Fri Jul  1 07:18:59 2016 (05:05:19 ago)
13|   Config Parameters:
14|     Bandwidth:        0 kbps (CT0) Priority:  7  7 Affinity:
0x0/0x0
15|     Metric Type: TE (global)
16|     Path Selection:
17|       Tiebreaker: Min-fill (default)
18|       Protection: any (default)
19|     Hop-limit: disabled
20|     Cost-limit: disabled
21|     Path-invalidation timeout: 10000 msec (default), Action:
Tear (default)
22|     AutoRoute: disabled  LockDown: disabled   Policy class:
not set
23|     Forward class: 0 (default)
24|     Forwarding-Adjacency: disabled
25|     Autoroute Destinations: 0
26|     Loadshare:          0 equal loadshares
27|     Auto-bw: disabled
```

```
    Path Protection: Not Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled
    SRLG discovery: Disabled
  History:
    Tunnel has been up for: 03:33:43 (since Fri Jul 01 08:50:35 UTC 2016)
    Current LSP:
      Uptime: 03:33:43 (since Fri Jul 01 08:50:35 UTC 2016)

  Segment-Routing Path Info (IGP information is not used)
    Segment0[First Hop]: 99.1.5.5, Label: -
    Segment1[ - ]: Label: 16004
    Segment2[ - ]: Label: 30403
    Segment3[ - ]: Label: 16008
    Segment4[ - ]: Label: 16010
Displayed 1 (of 4) heads, 0 (of 0) midpoints, 0 (of 0) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

IGP shares the same SRTE Policies to protect multiple destinations; those that require the same repair path. This is illustrated in Example 9-4. ISIS uses tunnel-te32784 for the repair path of destination 1.1.1.9/32, 1.1.1.10/32, …, as shown in line 9 of the output.

Example 9-4: Sharing SRTE Policy to protect multiple destinations

```
 1 RP/0/0/CPU0:xrvr-1#show isis fast-reroute tunnel
 2
 3 IS-IS 1 SRTE backup tunnels
 4
 5 tunnel-te32784, state up
 6   Outgoing interface: GigabitEthernet0/0/0/0
 7   Next hop: 99.1.5.5
 8   Label stack: 16004, 24002, 16008
 9   Prefix: 1.1.1.9/32 1.1.1.10/32 99.2.10.0/24 99.8.9.0/24 99.9.10.0/24
10
11 tunnel-te32783, state up
12   Outgoing interface: GigabitEthernet0/0/0/0
13   Next hop: 99.1.5.5
14   Label stack: 16004, 24002, 16008, 16010
```

```
   Prefix: 1.1.1.6/32 99.6.10.0/24
```

9.5 TI-LFA Protection Options

In this section, we will look at the different protection options that are available for TI-LFA, how they are configured for both ISIS and OSPF. We will also see examples of each using the network topology of Figure 9-11, which is used to illustrate TI-LFA for both ISIS and OSPF.

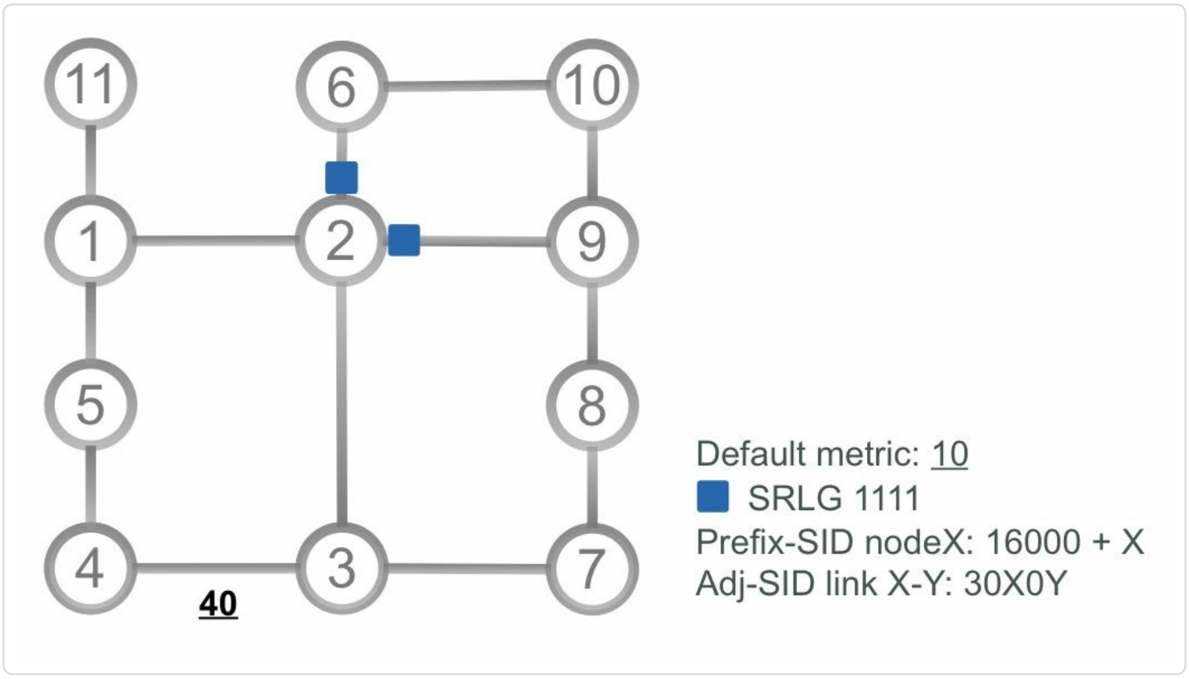


Figure 9-11: TI-LFA network topology

All links have an IGP metric 10, except the link between Node3 and Node4 that has an IGP metric 40. All nodes have allocated the default SRGB [16000-23999]. Each NodeX has a loopback prefix 1.1.1.X/32 and an associated Prefix-SID label 16000 + X. Node2 has two links that share an SRLG “1111”: link to Node6 and link to Node9. These links are marked with a square in the illustration.

HIGHLIGHT

TI-LFA protection options all ensure a repair path along the post-convergence path in any topology

The default link protection option works in many network designs and is preferred; it can even provide de facto node protection in many cases

SRLG protection can be enabled in networks where SRLGs are provisioned; note that SRLGs for only local links are considered in the implementation in Cisco IOS XR at the time of writing this book

Node protection may be useful to enable in certain cases but it might not provide the post-convergence path if the failure was just a link failure

9.5.1 Link Protection

Link protection is the most basic of the options that is available for TI-LFA and all it needs is to be just enabled for all the links that need protection.

9.5.1.1 ISIS Configuration

Example 9-5 shows the very simple ISIS configuration to enable TI-LFA Link Protection. `fast-reroute per-prefix` and `fast-reroute per-prefix ti-lfa` are enabled under the address-family of each ISIS interface. TI-LFA is supported for both IPv4 and IPv6 address-families.

Example 9-5: ISIS TI-LFA link protection

```
router isis 1
 address-family ipv4 unicast
  segment-routing mpls
 !
 address-family ipv4 unicast
  segment-routing mpls
 !
 interface GigabitEthernet0/0/0/0
```

```
   point-to-point
   address-family ipv4 unicast
    fast-reroute per-prefix
    fast-reroute per-prefix ti-lfa
   !
   address-family ipv6 unicast
    fast-reroute per-prefix
    fast-reroute per-prefix ti-lfa
```

To automatically add the TI-LFA configuration to all ISIS interfaces, one can use the configuration group functionality; see Example 9-6. The configuration group functionality is a generic configuration simplification tool, which is not specific to TI-LFA or Segment Routing.

Example 9-6: ISIS TI-LFA link protection – using configuration group

```
group GROUP_ISIS
 router isis '.*'
  interface 'GigabitEthernet.*'
   address-family ipv4 unicast
    fast-reroute per-prefix
    fast-reroute per-prefix ti-lfa
   address-family ipv6 unicast
    fast-reroute per-prefix
    fast-reroute per-prefix ti-lfa
end-group
!
router isis 1
 apply-group GROUP_ISIS
 address-family ipv4 unicast
  segment-routing mpls
 !
 address-family ipv4 unicast
  segment-routing mpls
!
 interface GigabitEthernet0/0/0/0
  point-to-point
  address-family ipv4 unicast
```

9.5.1.2 OSPF Configuration

Example 9-7 shows a TI-LFA configuration example for OSPF. The TI-LFA configuration for OSPF has interface scope. It is applicable per interface but it can be configured per instance, per area, and per interface. The regular OSPF configuration inheritance rules are applied on the configuration: if TI-LFA is configured per instance or per area, then all interfaces in the instance or area inherit the TI-LFA configuration.

Example 9-7: OSPF TI-LFA configuration

```
router ospf 1
 segment-routing mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
```

Example 9-8 illustrates how fast-reroute and/or TI-LFA can be disabled per area or per interface.

Example 9-8: OSPF TI-LFA configuration – disabling

```
router ospf 1
 segment-routing mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 area 0
  interface GigabitEthernet0/0/0/0
   network point-to-point
   fast-reroute disable
  !
 !
 area 1
  fast-reroute per-prefix ti-lfa disable
  interface GigabitEthernet0/0/0/1
   network point-to-point
  !
 !
!
```

9.5.1.3 Link Protection Examples

A single TI-LFA link-protecting example is discussed here: a double-segment link protecting repair path.

Refer to the network topology in Figure 9-12.

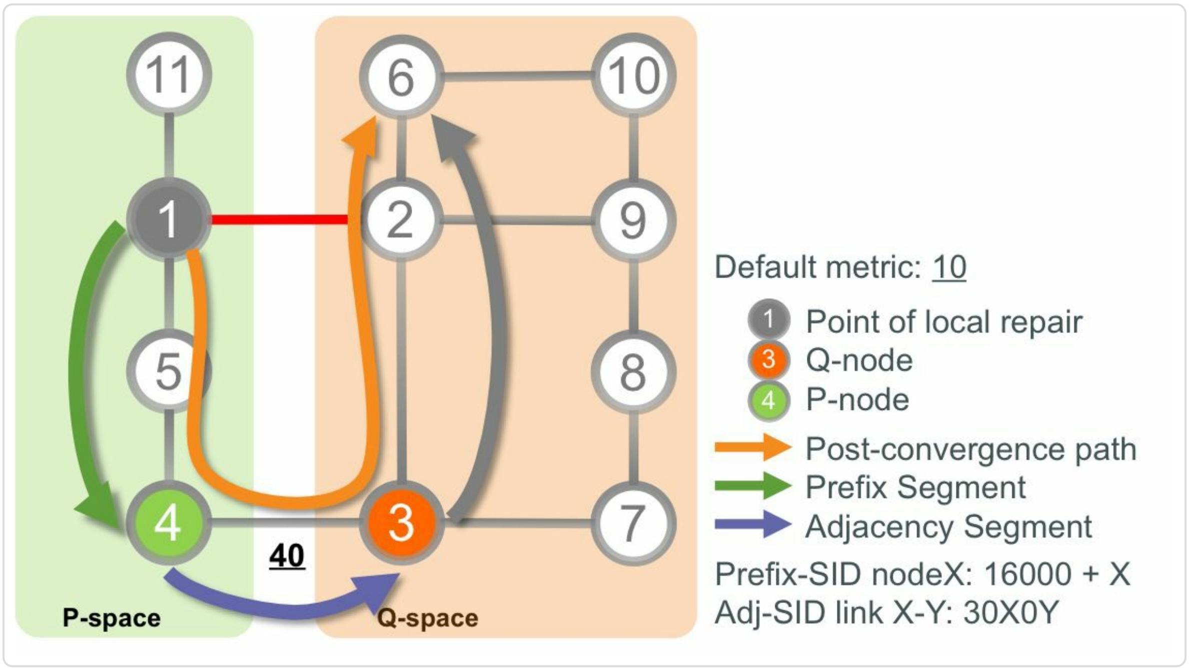


Figure 9-12: TI-LFA double-segment link protection

For Node1, the primary path towards destination Node6 traverses the link between Node1 and Node2. The TI-LFA computation for destination Node6 is as follows:

- Remove the link on the primary path to Node6 (link Node1-Node2) from the topology and compute the Shortest Path Tree (SPT) on the resulting topology. This gives us the post-convergence path from Node1 to Node6: {Node5, Node4, Node3, Node2, Node6}. This path is labeled “Post-convergence path” in the illustration.
- Node4 is in the P-space (Node1 can send a packet destined to Node4 without any risk of having that packet flow back through the protected link Node1-Node2). Node4 is not in the Q space; a packet destined for

Node6 that is steered into Node4 is looped back and sent over the protected link. The P-space and Q-space are indicated in the illustration.

- Node3 is in the Q-space (Node3 can send a packet to Node2 without any risk of having this packet flow back through the protected link Node1-Node2).
- The P- and Q-spaces do not intersect; there is no node that is in both P-space and Q-space. But Node4 (P-node) and Node3 (Q-node) are adjacent and located on the post-convergence path.
- The PLR Node1 pushes the segment list {Prefix-SID(Node4), Adjacency-SID(R4-R3)} on the repaired packets to destination Node6 and forwards them on interface to Node5. If the shortest path from Node4 to Node3 is via the direct link between these nodes, then the Prefix-SID of Node4 can be used to steer the traffic from Node3 to Node4. Otherwise the Adjacency-SID of the link Node4-Node3 is used. In this example the latter case is illustrated: the shortest path from Node4 to Node3 consists of two equal cost paths (via Node5 and directly to Node3). Also in that case the Adjacency-SID is used to steer the traffic to Node3.

The above behavior is applied per-prefix. The post-convergence path is computed for each destination prefix, and the TI-LFA repair path for each destination is then tailored over the post-convergence path to that destination. Note that the above description is a conceptual description of the computations instead of the actual algorithm. The actual TI-LFA algorithm is proprietary (local behavior which is not in the scope of IETF standardization) and scales extremely well.

Figure 9-13 illustrates the label stack on the packets directed on the TI-LFA repair path after the failure of the link between Node1 and Node2. Node6 advertises a loopback prefix 1.1.1.6/32 with a Prefix-SID label 16006.

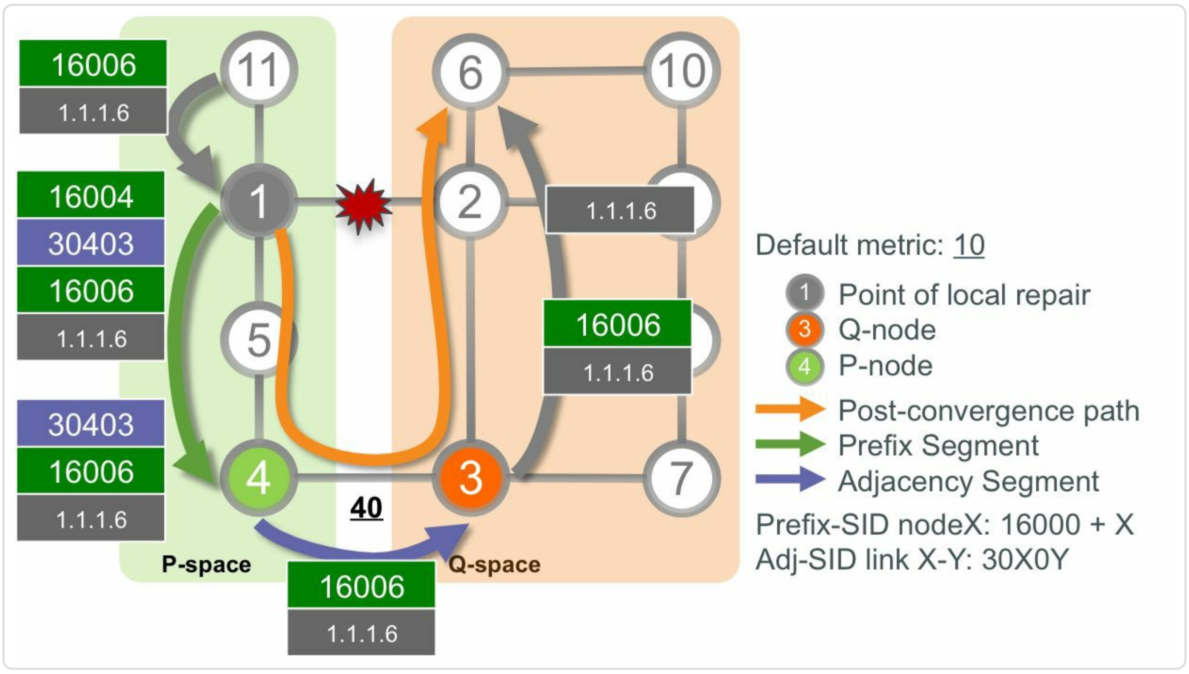


Figure 9-13: Double-segment repair path, label stack

A packet with top label 16006 arrives at Node1. 16006 is a Prefix-SID of Node6. Node1 swaps the top label with the same Prefix-SID label 16006. Then Node1 pushes two additional labels on the label stack: the first label is the Prefix-SID label 16004 associated with the loopback prefix of the P-node Node4. The second label is the Adjacency-SID label associated with the adjacency of Node4 to Node3. This is a label locally allocated by Node4, for example 30403 as used in the illustration. Then Node1 directs the packet on the interface towards Node4.

If an unlabeled IP packet with destination 1.1.1.6 arrives at Node1, then Node1 pushes the Prefix-SID label of 1.1.1.6/32, 16006. This is equivalent to the Prefix-SID label imposition on the primary path when the repair path is not active. Then Node1 pushes the same two additional labels on the label stack as described above: the Prefix-SID label 16004 associated with the loopback prefix of the P-node Node4, and the Adjacency-SID label 30403 associated with the adjacency of Node4 to Node3.

Node3 is in the Q-space of destination Node6. Thus when the packet on the repair path arrives on Node3, then Node3 forwards the packet on its regular shortest path towards destination Node6.

Example 9-9 shows the ISIS computed link protecting repair path in the output of `show isis fast-reroute detail`. This TI-LFA repair path uses Node4 as P-node and Node3 as Q-node. Traffic on the backup path is directed on the outgoing interface Gi0/0/0/0 towards Node5, nexthop 99.1.5.5 (line 6). ISIS has composed the label stack [16004, 30403, 16006] for the repair path. The bottom label 16006 is the Prefix-SID label associated with prefix 1.1.1.6/32 (line 8). This label is not an additional imposed label, but the regular Prefix-SID outgoing label for the destination. The next label on the stack, in the middle, is the Adjacency-SID label 30403 for the adjacency from Node4 to Node3 (line 7). The top label 16004 is the Prefix-SID label for the P-node, Node4, associated with the prefix 1.1.1.4/32 (line 6).

Example 9-9: ISIS double-segment repair path

```
1 RP/0/0/CPU0:xrvr-1#show isis fast-reroute 1.1.1.6/32 detail
2
3 L2 1.1.1.6/32 [20/115] medium priority
4      via 99.1.2.2, GigabitEthernet0/0/0/1, xrvr-2, SRGB
Base: 16000, Weight: 0
5          Backup path: TI-LFA (link), via 99.1.5.5,
GigabitEthernet0/0/0/0 xrvr-5, SRGB Base: 16000, Weight: 0
6            P node: xrvr-4.00 [1.1.1.4], Label: 16004
7            Q node: xrvr-3.00 [1.1.1.3], Label: 30403
8            Prefix label: 16006
```

```
          P: No, TM: 80, LC: No, NP: No, D: No, SRLG: No
        src xrvr-6.00-00, 1.1.1.6, prefix-SID index 6, R:0 N:1
P:0 E:0 V:0 L:0
```

The Adjacency-SID to steer the protected traffic from Node4 to Node3 is the unprotected Adjacency-SID. The output collected on the P-node Node4 in Example 9-10 shows the unprotected Adjacency-SID to the Q-node Node3 with label 30403. This Adjacency-SID label is used in the TI-LFA repair path.

Example 9-10: ISIS Adjacency-SID from P-node to Q-node

```
RP/0/0/CPU0:xrvr-4#show isis adjacency systemid xrvr-3 detail

IS-IS 1 Level-2 adjacencies:
System Id      Interface        SNPA           State Hold
Changed  NSF IPv4 IPv6

 BFD  BFD
xrvr-3         Gi0/0/0/1        PtoP          Up    28
1w0d     Yes None None
  Area Address:           49.0001
  Neighbor IPv4 Address:  99.3.4.3*
  Adjacency SID:          310403 (protected)
   Backup label stack:    [16003]
   Backup stack size:     1
   Backup interface:      Gi0/0/0/0
   Backup nexthop:        99.4.5.5
   Backup node address:   1.1.1.3
  Non-FRR Adjacency SID:  30403
  Topology:               IPv4 Unicast

Total adjacency count: 1
```

Example 9-11 shows the OSPF calculated repair path in the output of `show ospf routes backup-path`. The TI-LFA repair path is a path via P-node Node4, with label 16004 (line 10), and Q-node Node3 with label 30403 (line 11). Traffic on the backup path is directed on the outgoing interface Gi0/0/0/0 towards Node5, nexthop 99.1.5.5 (line 12).

Example 9-11: OSPF double-segment repair path

```
 1 RP/0/0/CPU0:xrvr-1#show ospf route 1.1.1.6/32 backup-path detail
 2
 3 OSPF Route entry for 1.1.1.6/32
 4   Route type:  Intra-area
 5   Last updated: Jun 30 19:58:13.752
 6   Metric: 21
 7     SPF priority: 4,  SPF version: 37
 8   RIB version: 0,  Source: Unknown
 9        99.1.2.2, from 1.1.1.6, via GigabitEthernet0/0/0/1, path-id 1
10            Backup path: TI-LFA, Repair-List: P node: 1.1.1.4         Label: 16004
11                                              Q node: 1.1.1.3         Label: 30403
12             99.1.5.5, from 1.1.1.6, via GigabitEthernet0/0/0/0, protected bitmap 0000000000000001
13             Attributes: Metric: 81, SRLG Disjoint
```

The output on P-node Node4 in Example 9-12 shows the unprotected Adjacency-SID to the Q-node with label 30403 (line 23). This Adjacency-SID label is used in the TI-LFA repair path.

Example 9-12: OSPF Adjacency-SID from P-node to Q-node

```
 1 RP/0/0/CPU0:xrvr-4#show ospf neighbor det 1.1.1.3
 2
 3 * Indicates MADJ interface
 4 # Indicates Neighbor awaiting BFD session up
 5
 6 Neighbors for OSPF 1
 7
 8  Neighbor 1.1.1.3, interface address 99.3.4.3
 9     In the area 0 via interface GigabitEthernet0/0/0/1
10     Neighbor priority is 1, State is FULL, 6 state changes
11     DR is 0.0.0.0 BDR is 0.0.0.0
```

```
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:00:33
    Neighbor is up for 1w0d
    Number of DBD retrans during last exchange 0
    Index 2/2, retransmission queue length 0, number of retransmission 10
    First 0(0)/0(0) Next 0(0)/0(0)
    Last retransmission scan length is 0, maximum is 2
    Last retransmission scan time is 0 msec, maximum is 0 msec
    LS Ack list: NSR-sync pending 0, high water mark 0
    Adjacency SID Label: 310403, Protected
    Unprotected Adjacency SID Label: 30403
```

To verify the forwarding behavior for incoming unlabeled packets, the cef table is examined. The output of `show cef` for prefix 1.1.1.6/32 in Example 9-13 shows a primary path via Node2, nexthop 99.1.2.2, with a `path-idx 1` (lines 11-14). The label 16006 is imposed on the primary path; this is the Prefix-SID label of the destination Node6. The primary path is flagged as `protected` by the path with path-idx 0, indicated by `bkup-idx 0`.

The repair path (lines 6-10 in the output) has `path-idx 0` and goes via Node5, nexthop 99.1.5.5, using P-node 1.1.1.4 and Q-node 1.1.1.3. The imposed label stack contains three labels: {16004, 30403, 16006}. The bottom label, on the right, is the Prefix-SID label 16006 associated with destination prefix 1.1.1.6/32. This is the regular Prefix-SID label that is also imposed on packets destined for 1.1.1.6 on the primary path when the repair path is not active. The next label on the stack, in the middle, is the Adjacency-SID label 30403 for the adjacency from Node4 to Node3. The top label 16004 on the left, is the Prefix-SID label for the P node Node4, associated with the prefix 1.1.1.4/32.

Example 9-13: TI-LFA double-segment repair path in CEF

```
RP/0/0/CPU0:xrvr-1#show cef 1.1.1.6/32
1.1.1.6/32, version 1214, internal 0x1000001 0x81 (ptr 0xa1434d74) [1], 0x0 (0xa13ffb00), 0xa28 (0xa171c470)
 Updated Jun 30 21:42:14.375
 local adjacency 99.1.2.2
 Prefix Len 32, traffic index 0, precedence n/a, priority 1
   via 99.1.5.5/32, GigabitEthernet0/0/0/0, 11 dependencies, weight 0, class 0, backup (remote) [flags 0x8300]
    path-idx 0 NHID 0x0 [0xa110d250 0x0]
    next hop 99.1.5.5/32, P-node 1.1.1.4, Q-node 1.1.1.3
    local adjacency
     local label 16006      labels imposed {16004 30403 16006}
   via 99.1.2.2/32, GigabitEthernet0/0/0/1, 11 dependencies, weight 0, class 0, protected [flags 0x400]
    path-idx 1 bkup-idx 0 NHID 0x0 [0xa0e834e0 0x0]
    next hop 99.1.2.2/32
     local label 16006      labels imposed {16006}
```

To verify the forwarding behavior for incoming SR labeled packets, the MPLS forwarding table is examined. See the output of `show mpls forwarding detail` for Prefix-SID label 16006 in Example 9-14.

The MPLS forwarding entry on top of the output (lines 5-12) shows the primary path via Node2, nexthop 99.1.2.2. The local or incoming label 16006 is swapped with the same outgoing label 16006. The primary path is marked as protected by the path with index 0, as indicated by `Backup path idx: 0` on line 10.

On the repair path, the same label stack is shown as in the cef entry: {16004, 30403, 16006} (line 18). Note that the full label stack is only shown in the detailed output. The regular, non-detailed output only shows the top label, 16004 in this example (as in line 14). The repair path goes via outgoing interface Gi0/0/0/0, nexthop Node5.

Example 9-14: TI-LFA double-segment protection MPLS forwarding

```
RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 16006 detail
Local  Outgoing    Prefix             Outgoing     Next Hop
    Bytes
Label  Label       or ID
Interface                    Switched
------ ----------- ------------------ ------------ ---------
------ ------------
16006  16006       SR Pfx (idx 6)     Gi0/0/0/1
99.1.2.2        0
     Updated: Jun 30 21:42:12.444
     Path Flags: 0x400 [  BKUP-IDX:0 (0xa0e834e0) ]
     Version: 1214, Priority: 1
     Label Stack (Top -> Bottom): { 16006 }
     NHID: 0x0, Encap-ID: N/A, Path idx: 1, Backup path idx:
0, Weight: 0
     MAC/Encaps: 14/18, MTU: 1500
     Packets Switched: 0

       16004       SR Pfx (idx 6)     Gi0/0/0/0
99.1.5.5        0            (!)
     Updated: Jun 30 21:42:12.444
     Path Flags: 0x8300 [  IDX:0 BKUP, NoFwd ]
     Version: 1214, Priority: 1
     Label Stack (Top -> Bottom): { 16004 30403 16006 }
     NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx:
0, Weight: 0
     MAC/Encaps: 14/26, MTU: 1500
     Packets Switched: 0
     (!): FRR pure backup

  Traffic-Matrix Packets/Bytes Switched: 0/0
```

9.5.2 Node and SRLG Protection

TI-LFA also protects against node and local SRLG failures. Since TI-LFA installs a single repair path per destination, the operator specifies a preference order to indicate which component should be protected: link, node, or SRLG. If no preference is configured then only link protection is considered. The repair path of the preferred protection mode is computed and installed. This programmed repair path is then activated upon failure, regardless if the failure is a link, node, or SRLG failure. For example if TI-LFA installs a node protecting repair path, then that repair path is used upon any failure of the link, even if the node stays up.

The configuration of the preference order is done using a tiebreaker configuration. Although the configuration syntax is leveraged from Classic LFA, the semantic is slightly different. In Classic LFA the tiebreaker rules are used to select one LFA among multiple candidate LFAs. The tiebreaker rules as used in TI-LFA do not select among multiple LFA candidates, but specify a protection mode (link, node, SRLG) preference order. To start with, the candidate must be a post convergence path since this is the fundamental property of TI-LFA. If the repair path computation of a higher preference protection mode does not yield a repair path then the path computation of the next lower preference protection mode is tried. TI-LFA tries to combine all configured protection modes. If both SRLG and node protection are configured, then TI-LFA tries to provide a combined node + SRLG protection. Node and SRLG protection each implicitly provides link protection as well.

For example, if the preference order as configured with the tiebreakers is: (1) local SRLG protection, (2) node protection, (3) link protection, then TI-LFA prefers an SRLG + node protecting repair path. If no SRLG + node protecting repair path is found then TI-LFA prefers an SRLG protecting repair path. If that is not available, then TI-LFA prefers a node protecting repair path. If that is not available either then TI-LFA uses a link protecting repair path. With TI-LFA, as long as there is an alternate path, the algorithm always finds a repair path that is link protecting.

9.5.2.1 ISIS Configuration

ISIS allows configuration of per address-family tiebreaker rules under the ISIS instance and under the ISIS interface. Any tiebreaker configuration under an ISIS interface completely overrules the tiebreaker configuration under the ISIS instance for that address-family. The instance address-family tiebreaker configuration is completely ignored, i.e. there is no merge between interface and instance tiebreaking rules. This is in contrast with OSPF where the tiebreakers configured at different levels are merged; see section 9.5.2.2, "OSPF Configuration".

The tiebreaker rules can be configured per ISIS level.

TI-LFA can use the three tiebreaker types (`node-protecting`, `srlg-disjoint`, and `default`) shown in Example 9-15. The other (classic LFA) tiebreakers are not supported for TI-LFA in ISIS. The first two types (`node-protecting` and `srlg-disjoint`) are used to indicate preference for node protection and SRLG protection. The third one (`default`) is somewhat special. It is used to indicate that the default protection must be used, which is link protection. It can be used to revert to the default protection for a given ISIS interface when some node and/or SRLG protection preference is configured under the ISIS instance. The examples below will clarify this more.

Example 9-15: ISIS TI-LFA tiebreaker rules

```
fast-reroute per-prefix tiebreaker node-protecting index
<index> {level <level>}
fast-reroute per-prefix tiebreaker srlg-disjoint index <index>
{level <level>}
fast-reroute per-prefix tiebreaker default {level <level>}
```

The `index` indicates the preference of the rule. A higher index means a more preferred tiebreaker.

The `default` tiebreaker is mutually exclusive with the `node-protecting` and `srlg-disjoint` tiebreakers; they cannot be configured together on the same ISIS level.

The following examples illustrate some possible configurations.

In Example 9-16, TI-LFA tiebreaker rules are configured under the ISIS instance's address-family (lines 3 to 4). No tiebreaker is configured under the ISIS interface, hence the interface inherits the tiebreakers configured under the instance. With this configuration, TI-LFA provides a node + SRLG protecting repair path for destinations reachable via interface Gi0/0/0/0 if available. This repair path would also be link protecting. If no such repair path were available, a node-protecting repair path would be selected since the node-protecting tiebreaker rule has higher preference (higher index). If no such repair path is available then TI-LFA selects an SRLG-disjoint repair path. If that is not available then TI-LFA selects a link-protecting repair path. Note that the implicit link-protection tiebreaker rule is always at the lowest preference level (not configurable).

Example 9-16: ISIS TI-LFA node and SRLG protection

```
1 router isis 1
2  address-family ipv4 unicast
3   fast-reroute per-prefix tiebreaker node-protecting index 200
4   fast-reroute per-prefix tiebreaker srlg-disjoint index 100
5  !
6  interface GigabitEthernet0/0/0/0
7   address-family ipv4 unicast
8    fast-reroute per-prefix
9    !!! no tiebreaker
```

```
10    fast-reroute per-prefix ti-lfa
```

In Example 9-17 a tiebreaker rule is configured under the ISIS instance (line 3) and one under the ISIS interface (line 8). The interface tiebreaker configuration completely overrules the instance tiebreaker configuration. As indicated before, the different tiebreaker rules are not merged. With this configuration TI-LFA selects an SRLG-disjoint repair path for destinations reachable via primary interface Gi0/0/0/0. If no such repair path is available then TI-LFA selects a link protecting repair path.

Example 9-17: ISIS TI-LFA configuration on instance and interface level

```
1 router isis 1
2  address-family ipv4 unicast
3   fast-reroute per-prefix tiebreaker node-protecting index
100
4  !
5  interface GigabitEthernet0/0/0/0
6   address-family ipv4 unicast
7    fast-reroute per-prefix
8    fast-reroute per-prefix tiebreaker srlg-disjoint index 200
9    fast-reroute per-prefix ti-lfa
```

Example 9-18 shows an example where tiebreakers are configured under the ISIS instance (lines 3 and 4) and the default tiebreaker is configured under the ISIS interface (line 10). With this configuration TI-LFA computes a link-protecting repair path for destinations reachable via primary interface Gi0/0/0/0 since the interface tiebreaker configuration completely overrules the ISIS instance tiebreakers.

Example 9-18: ISIS TI-LFA configuration – revert to default on interface

```
1 router isis 1
2  address-family ipv4 unicast
3   fast-reroute per-prefix tiebreaker node-protecting index
100
```

```
 fast-reroute per-prefix tiebreaker srlg-disjoint index 200
!
interface GigabitEthernet0/0/0/0
 address-family ipv4 unicast
  fast-reroute per-prefix
  fast-reroute per-prefix ti-lfa
  fast-reroute per-prefix tiebreaker default
```

9.5.2.2 OSPF Configuration

OSPF allows configuration of tiebreaker rules under the OSPF instance and under the OSPF interface. In contrast to ISIS, OSPF merges tiebreaker rules configured under the OSPF instance with tiebreaker rules configured under an OSPF interface.

Each tiebreaker can be disabled or enabled with a specific `index`. The `index` indicates the preference of the rule. A higher index means a more preferred tiebreaker.

Some of the existing tiebreakers in OSPF are also applicable to TI-LFA protection. See Table 9-1. The first column in this table specifies the tiebreaker type in the configuration command. The second column contains a description of the tiebreaker. The third column indicates if the tiebreaker is applicable for TI-LFA, and the fourth column indicates the default preference index, or “Disabled” if the tiebreaker is disabled by default. If the configured tiebreaker is applicable to TI-LFA and has a higher preference index than the node or SRLG tiebreakers, then it is considered first. Tiebreakers that do not apply to TI-LFA are ignored, regardless of their preference. Note that the most preferred tiebreaker for TI-LFA is “post-convergence”. This tiebreaker is not configurable and indicates the preference for the repair path to follow the post-convergence path.

Table 9-1: OSPF fast-reroute tiebreakers

Tiebreaker	**Description**	**TI-LFA?**	**Default index**
`downstream`	Repair path is via the neighbor whose metric to the destination is lower than the metric of the protecting node (PLR) to the same destination.	No (only for per-prefix direct LFA)	Disabled
`lc-disjoint`	Repair path is using interface that is on a different Line-card (LC).	Yes	Disabled
`lowest-backup-metric`	Repair path uses the lowest metric. For direct LFA, the metric represents a distance to the protected prefix over the computed repair path. For RLFA the metric represents the metric to the PQ node. For TI-LFA the metric represents the metric to the prefix over the post-convergence repair path.	Yes	20
`node-protecting`	Repair path avoids the node that is being used as a next-hop for primary path. For TI-LFA, it enables the computation of candidate repair paths excluding the peer node.	Yes	Disabled
`primary-path`	Repair path is equal to one of the ECMP paths.	No (only for per-prefix direct LFA)	10
`secondary-path`	Repair path is not an ECMP path.	No (only for per-prefix direct LFA)	Disabled

`srlg-disjoint`	Repair path avoids the local links that share the same SRLG with the primary link. For TI-LFA, it enables the computation of candidate repair paths excluding all local links that share SRLG with the primary link.	Yes	Disabled

The OSPF LFA tiebreakers can be verified using the `show ospf interface` command. See Example 9-19. No tiebreakers were configured for this example. The list of tiebreakers (lines 21 to 29) contains two non-configurable tiebreakers: `No Tunnel (Implicit)` and `Post Convergence Path`. The former indicates that OSPF always gives higher preference to a non-tunnel interface repair path over a tunnel interface backup path (this is not related to the automatic instantiation of an SRTE Policy for the repair path as described in section 9.4.3, "TE Infrastructure for Repair Paths"). The latter indicates the higher preference for a repair path tailored over the post-convergence path. Both these tiebreakers have a preference higher than 255 and are therefore always preferred over configurable tiebreakers which have at most preference 255.

Example 9-19: Verify OSPF IPFRR tiebreakers

```
 1 RP/0/0/CPU0:xrvr-1#show ospf interface gigabitEthernet
0/0/0/0
 2
 3 GigabitEthernet0/0/0/0 is up, line protocol is up
 4   Internet Address 99.1.2.1/24, Area 0
 5   Process ID 1, Router ID 1.1.1.1, Network Type
POINT_TO_POINT, Cost: 1
 6   Transmit Delay is 1 sec, State POINT_TO_POINT, MTU 1500,
MaxPktSz 1500
 7   Timer intervals configured, Hello 10, Dead 40, Wait 40,
Retransmit 5
 8     Hello due in 00:00:05:029
 9   Index 1/1, flood queue length 0
10   Next 0(0)/0(0)
```

```
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  LS Ack List: current length 0, high water mark 4
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 1.1.1.2
  Suppress hello for 0 neighbor(s)
  Multi-area interface Count is 0
  Fast-reroute type Per-prefix
   IPFRR per-prefix tiebreakers:
    Name                        Index
    No Tunnel (Implicit)        257
    Lowest Metric               20
    Primary Path                10
    Post Convergence Path       256
    Downstream                  0
    Line-card Disjoint          0
    Node Protection             0
    Secondary Path              0
    SRLG Disjoint               0
    Topology Independent LFA enabled
```

The following examples illustrate some possible configurations.

In Example 9-20, TI-LFA tiebreaker rules are configured under the OSPF instance (lines 6 and 7). No tiebreaker is configured under the OSPF interface. The interface inherits the tiebreakers configured under the instance. With this configuration, TI-LFA prefers a node + SRLG protecting repair path for destinations reachable via interface Gi0/0/0/0 if available. If no such repair path were available, a lower preference repair path would be selected. The resulting tiebreaker rules for interface Gi0/0/0/0 are shown in Example 9-21. Lines 5 and 6 indicate that the node protection and SRLG disjoint tiebreakers are enabled with their configured preference indexes.

Example 9-20: OSPF instance TI-LFA node and SRLG protection

```
1 RP/0/0/CPU0:xrvr-12#show running-config router ospf
2 router ospf 1
```

```
segment-routing mpls
fast-reroute per-prefix
fast-reroute per-prefix ti-lfa enable
fast-reroute per-prefix tiebreaker node-protecting index 200
fast-reroute per-prefix tiebreaker srlg-disjoint index 100
area 0
 interface GigabitEthernet0/0/0/0
  network point-to-point
 !
!
!
```

Example 9-21: OSPF instance TI-LFA node and SRLG protection – verification

```
RP/0/0/CPU0:xrvr-12#show ospf interface Gi0/0/0/0 | begin tiebreakers
   IPFRR per-prefix tiebreakers:
    Name                        Index
    No Tunnel (Implicit)        257
    Node Protection             200
    SRLG Disjoint               100
    Lowest Metric               20
    Primary Path                10
    Post Convergence Path       256
    Downstream                  0
    Line-card Disjoint          0
    Secondary Path              0
    Topology Independent LFA enabled
```

In Example 9-22 a tiebreaker rule is configured under the OSPF instance (line 6) and one under the OSPF interface (line 10). The tiebreaker rules are merged, as seen in lines 5 and 6 of Example 9-23. With this configuration TI-LFA selects a node + SRLG protecting repair path for destinations reachable via primary interface Gi0/0/0/0. If no such repair path is available then TI-LFA selects, in order of preference, a SRLG protecting repair path, a node protecting repair path, or a link protecting repair path.

Example 9-22: OSPF instance TI-LFA node protection and interface SRLG protection

```
RP/0/0/CPU0:xrvr-12#show running-config router ospf
router ospf 1
 segment-routing mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 fast-reroute per-prefix tiebreaker node-protecting index 100
 area 0
  interface GigabitEthernet0/0/0/0
   network point-to-point
   fast-reroute per-prefix tiebreaker srlg-disjoint index 200
  !
 !
!
```

Example 9-23: OSPF instance TI-LFA node protection and interface SRLG protection – verification

```
RP/0/0/CPU0:xrvr-12#show ospf interface Gi0/0/0/0 | begin
tiebreakers
   IPFRR per-prefix tiebreakers:
    Name                       Index
    No Tunnel (Implicit)       257
    SRLG Disjoint              200
    Node Protection            100
    Lowest Metric              20
    Primary Path               10
    Post Convergence Path      256
    Downstream                 0
    Line-card Disjoint         0
    Secondary Path             0
    Topology Independent LFA enabled
```

Example 9-24 shows a configuration example where tiebreakers are configured under the OSPF instance (lines 6 and 7) and disabled under the OSPF interface (lines 11 and 12). With this configuration TI-LFA computes a link-protecting repair path for destinations reachable via primary interface Gi0/0/0/0 according to the resulting tiebreaker rules for the interface. The resulting tiebreaker rules for the interface are shown in Example 9-25.

Example 9-24: OSPF instance TI-LFA protection disabled for interface

```
RP/0/0/CPU0:xrvr-12#show running-config router ospf
router ospf 1
 segment-routing mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 fast-reroute per-prefix tiebreaker node-protecting index 100
 fast-reroute per-prefix tiebreaker srlg-disjoint index 200
 area 0
  interface GigabitEthernet0/0/0/0
   network point-to-point
   fast-reroute per-prefix tiebreaker node-protecting disable
   fast-reroute per-prefix tiebreaker srlg-disjoint disable
  !
 !
!
```

Example 9-25: OSPF instance TI-LFA protection disabled for interface – verification

```
RP/0/0/CPU0:xrvr-12#show ospf interface Gi0/0/0/0 | begin tiebreakers
   IPFRR per-prefix tiebreakers:
    Name                       Index
    No Tunnel (Implicit)       257
    Lowest Metric              20
    Primary Path               10
    Post Convergence Path      256
    SRLG Disjoint              0
    Node Protection            0
    Downstream                 0
    Line-card Disjoint         0
    Secondary Path             0
    Topology Independent LFA enabled
```

9.5.2.3 Node Protection Example

TI-LFA node protection is illustrated using the network topology in Figure 9-14. Node1 is configured to protect the traffic to destination Node6 against node failures.

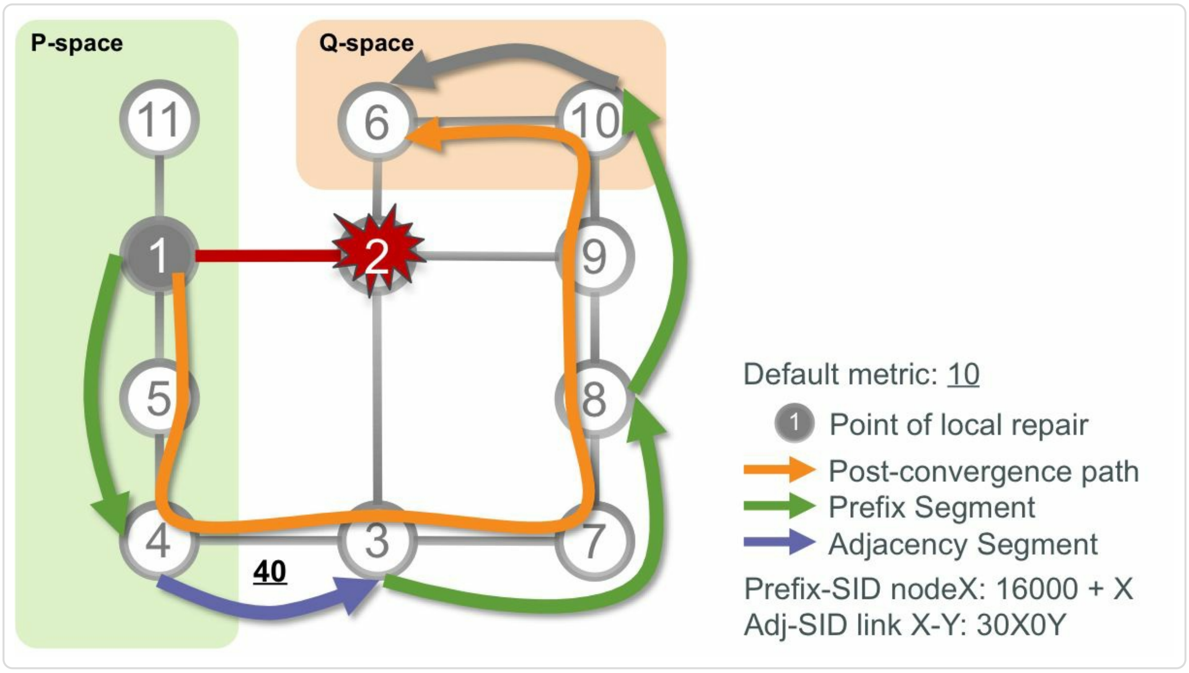


Figure 9-14: TI-LFA node protection

The primary path on Node1 towards destination Node6 traverses Node2. The TI-LFA computation for destination Node6 is as follows:

- Remove the next-hop node on the primary link to Node6 (Node2) from the topology and compute the SPT on the resulting topology. This gives us the post-convergence path from Node1 to Node6: {Node5, Node4, Node3, Node7, Node8, Node9, Node10, Node6}. This path is labeled "Post-convergence path" in the illustration.
- Node4 is in the P-space (Node1 can send a packet destined to Node4 without any risk of having that packet flow back through the protected link Node1-Node2). Node4 is not in the Q space; a packet destined for Node6 that is steered into Node4 is looped back and sent over the

protected link. The P-space and Q-space are indicated in the illustration.

- Node10 is in the Q-space (Node10 can send a packet to Node6 without any risk of having this packet flow back through the protected Node2).
- The P- and Q-spaces do not intersect and are not adjacent; there is no node that is in both P-space and Q-space nor is there a pair of adjacent P- and Q-nodes. TI-LFA computes a loop-free path along the post-convergence path to bridge the gap between P-space and Q-space. In this example the path consists of one Adjacency Segment and two Prefix Segments. The Adjacency Segment brings the packets from Node4 to Node3. The Prefix Segments then steer the packets on a loop-free path to Node10 via Node8. The shortest paths from Node3 to Node8 and from Node8 to Node10 do not traverse Node2 and are along the post-convergence path.
- The PLR Node1 pushes the segment list {Prefix-SID(Node4), Adjacency-SID(R4-R3), Prefix-SID(Node8), Prefix-SID(Node10)} on the repaired packets to destination Node6 and forwards them on interface to Node5.

The above behavior is applied per-prefix. The post-convergence path is computed for each destination prefix, and the TI-LFA repair path for each destination is then tailored over the post-convergence path to that destination. Note that the above description is a conceptual description of the computations instead of the actual algorithm. The actual TI-LFA algorithm is proprietary (local behavior which is not in the scope of IETF standardization) and scales extremely well.

Example 9-26 shows the ISIS computed TI-LFA repair path on Node1 for the topology in Figure 9-14. The output shows that it is a node protection repair path (line 5). The segments of the repair path are as illustrated in Figure 9-14. Notice that the Prefix Segments in the list are labeled `P node` (lines 7, 9-10) and the Adjacency Segment is labeled `Q node` (line 8). A P-node can be reached from the previous node without traversing the protected node, while a Q-node can reach the next node on the path without traversing the protected node. The repair path Segment List contains more than three segments, therefore ISIS dynamically instantiated a SRTE Policy, tunnel-te32783 in this example (line 6). This involvement of the TE infrastructure is discussed in more detail in section 9.4.3, "TE Infrastructure for Repair Paths".

Example 9-26: ISIS TI-LFA node protecting repair path

```
 1| RP/0/0/CPU0:xrvr-1#show isis fast-reroute 1.1.1.6/32 detail
 2|
 3| L2 1.1.1.6/32 [20/115] medium priority
 4|      via 99.1.2.2, GigabitEthernet0/0/0/1, xrvr-2, SRGB
Base: 16000, Weight: 0
 5|          Backup path: TI-LFA (node), via 99.1.5.5,
GigabitEthernet0/0/0/0 xrvr-5, SRGB Base: 16000, Weight: 0
 6|          Backup tunnel: tunnel-te32783
 7|            P node: xrvr-4.00 [1.1.1.4], Label: 16004
 8|            Q node: xrvr-3.00 [1.1.1.3], Label: 30403
 9|            P node: xrvr-8.00 [1.1.1.8], Label: 16008
10|            P node: xrvr-10.00 [1.1.1.10], Label: 16010
11|            Prefix label: 16006
12|        P: No, TM: 110, LC: No, NP: No, D: No, SRLG: No
13|      src xrvr-6.00-00, 1.1.1.6, prefix-SID index 6, R:0 N:1
P:0 E:0 V:0 L:0
```

The OSPF computed TI-LFA repair path on Node1 for the same topology is shown in Example 9-27. The path is node protecting (see line 16) and uses the segment list as illustrated in Figure 9-14. The nodes on the repair path are labeled with `P node` and `Q node` (lines 10-13), the same as for ISIS. OSPF instantiated a SRTE Policy since the repair path requires imposition of more than three labels, tunnel-te32781 in this example (line 14).

Example 9-27: OSPF TI-LFA node protecting repair path

```
 1 RP/0/0/CPU0:xrvr-1#show ospf routes 1.1.1.6/32 backup-path detail
 2
 3 OSPF Route entry for 1.1.1.6/32
 4   Route type:  Intra-area
 5   Last updated: Jun 30 19:58:13.752
 6   Metric: 21
 7     SPF priority: 4,  SPF version: 44
 8   RIB version: 0,  Source: Unknown
 9        99.1.2.2, from 1.1.1.6, via GigabitEthernet0/0/0/1, path-id 1
10          *Backup path: TI-LFA, Repair-List: *
11                                 P node: 1.1.1.4    Label: 16004
12                                 Q node: 1.1.1.3    Label: 30403
13                                 P node: 1.1.1.8    Label: 16008
14                                 P node: 1.1.1.10   Label: 16010
15                                 Backup Tunnel: tunnel-te32781
16                 99.1.5.5, from 1.1.1.6, via GigabitEthernet0/0/0/0, protected bitmap 0000000000000001
17                 Attributes: Metric: 111, Node Protect, SRLG Disjoint
```

The cef entry for Node6's loopback prefix 1.1.1.6/32 on Node1 is shown in Example 9-28. On the repair (backup) path, Node1 imposes Prefix-SID label 16006 (line 14) and steers into tunnel-te32781 (line 10).

Example 9-28: TI-LFA node protecting repair path in CEF

```
 1 RP/0/0/CPU0:xrvr-1#show cef 1.1.1.6/32
 2 1.1.1.6/32, version 1295, internal 0x1000001 0x81 (ptr 0xa1434d74) [1], 0x0 (0xa13ffb00), 0xa28 (0xa171c12c)
 3  Updated Jul  1 08:50:35.947
 4|  local adjacency 99.1.2.2
 5|  Prefix Len 32, traffic index 0, precedence n/a, priority 1
 6|    via 99.1.2.2/32, GigabitEthernet0/0/0/1, 6 dependencies,
weight 0, class 0, protected [flags 0x400]
 7|     path-idx 0 bkup-idx 1 NHID 0x0 [0xa0e84360 0x0]
 8|     next hop 99.1.2.2/32
 9|      local label 16006      labels imposed {16006}
10|    via 0.0.0.0/32, tunnel-te32781, 6 dependencies, weight 0,
class 0, backup [flags 0x300]
11|     path-idx 1 NHID 0x0 [0xa110d934 0x0]
12|     next hop 0.0.0.0/32
13|     local adjacency
14|      local label 16006      labels imposed {16006}
```

The MPLS forwarding entry for Prefix-SID label 16006 on Node1 is displayed in Example 9-29. The backup path is shown in lines 14 to 19. The outgoing label is 16006 and outgoing interface is tunnel-te32781 (line 14).

Example 9-29: TI-LFA node protecting repair path in MPLS

```
 1| RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 16006 detail
 2| Local  Outgoing    Prefix             Outgoing     Next
Hop          Bytes
 3| Label  Label       or ID
Interface                       Switched
 4| ------ ----------- ------------------ ------------ ---------
------ ------------
 5| 16006  16006       SR Pfx (idx 6)     Gi0/0/0/1
99.1.2.2        0
 6|       Updated: Jul  1 08:50:35.627
 7|       Path Flags: 0x400 [  BKUP-IDX:1 (0xa0e84360) ]
 8|       Version: 1295, Priority: 1
 9|       Label Stack (Top -> Bottom): { 16006 }
10|       NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx:
1, Weight: 0
11|       MAC/Encaps: 14/18, MTU: 1500
12|       Packets Switched: 0
13|
14|        16006       SR Pfx (idx 6)     tt32781
point2point     0           (!)
```

```
      Updated: Jul  1 08:50:35.627
      Path Flags: 0x300 [  IDX:1 BKUP, NoFwd ]
      Label Stack (Top -> Bottom): { }
      MAC/Encaps: 0/0, MTU: 0
      Packets Switched: 0
   Traffic-Matrix Packets/Bytes Switched: 0/0
```

The SRTE Policy forwarding entry for tunnel-te32781 on Node1 is displayed in Example 9-30. The outgoing interface and next-hop are in the output on line 5. The imposed stack of labels is listed on line 8.

Example 9-30: Automatic SRTE Policy forwarding entry

```
 1 RP/0/0/CPU0:xrvr-1#show mpls for tunnels 32781 detail
 2 Tunnel            Outgoing    Outgoing     Next Hop        Bytes
 3 Name              Label       Interface                    Switched
 4 ------------- ----------- ------------ --------------- ------------
 5 tt32781   (SR) 16004        Gi0/0/0/0    99.1.5.5        0
 6      Updated: Jul  1 08:50:35.607
 7      Version: 561, Priority: 2
 8      Label Stack (Top -> Bottom): { 16004 30403 16008 16010 }
 9      NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx: 0, Weight: 0
10      MAC/Encaps: 14/30, MTU: 1500
11      Packets Switched: 0
12
13   Interface Name: tunnel-te32781, Interface Handle: 0x00001280, Local Label: 24004
14   Forwarding Class: 0, Weight: 0
15   Packets/Bytes Switched: 0/0
```

9.5.2.4 SRLG Protection Example

TI-LFA SRLG protection is illustrated using the network topology in Figure 9-15. Node2 is the protecting node (PLR) in this example. Node2's interfaces to Node6 and Node9 share the same SRLG 1111. This SRLG is indicated in the illustration by a square on the link. Node2 is configured to protect the traffic to destination Node6 against SRLG failures. The implementation of Cisco IOS XR at the time of writing this book considers local SRLG failures: failures of remote links that are member of the same SRLG are not considered.

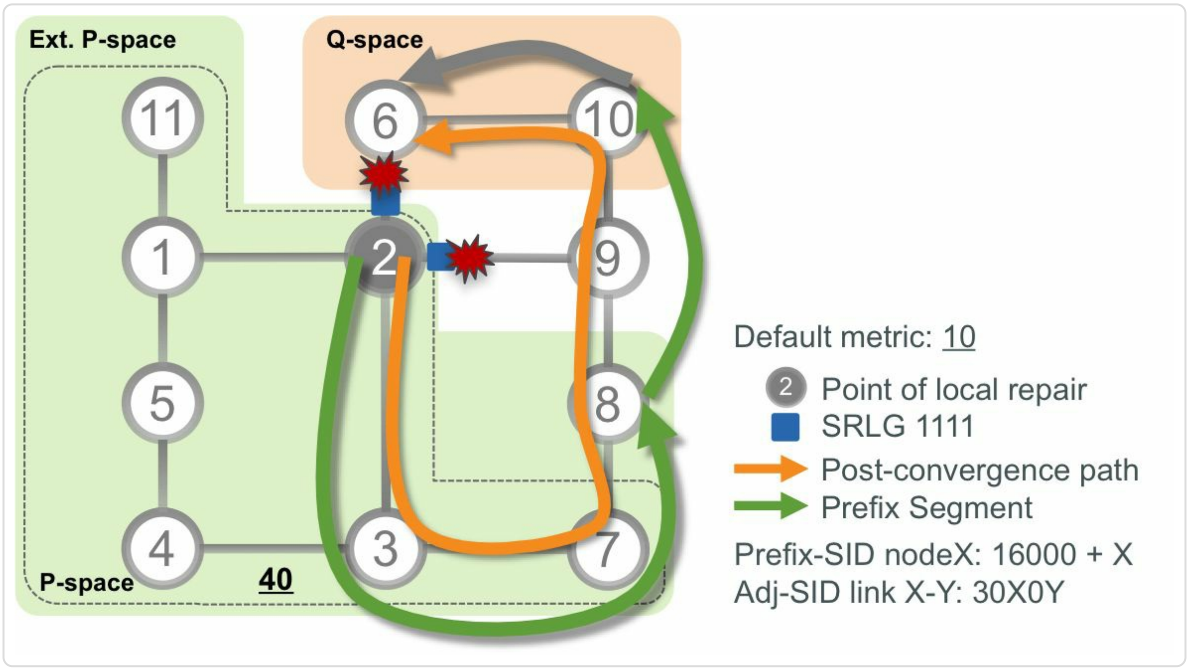


Figure 9-15: TI-LFA local SRLG protection

The primary path on Node2 towards destination Node6 is via the direct link between these two nodes. The TI-LFA computation on Node2 for destination Node6 is as follows:

- Remove the local links that share a same SRLG with the primary link to Node6 from the topology and compute the SPT on the resulting topology. The link to Node9 has the same SRLG as the link on the primary path (link to Node6). This gives us the post-convergence path from Node2 to Node6: {Node3, Node7, Node8, Node9, Node10, Node6}. This path is labeled “Post-convergence path” in the illustration.

- Node8 is in the Extended P-space (Node2 can send a packet destined to Node8 on the outgoing interface to Node3 without any risk of having that packet flow back through a link with a protected SRLG). Node8 is not in the Q space; a packet destined for Node6 that is steered into Node8 can traverse a link of the protected SRLG: link Node9-Node2. The (extended) P-space and Q-space are indicated in the illustration.
- Node10 is in the Q-space (Node10 can send a packet to Node6 without any risk of having this packet traverse a link with a protected SRLG).
- The P- and Q-spaces do not intersect and are not adjacent. TI-LFA computes a loop-free path along the post-convergence path to bridge the gap between P-space and Q-space. In this example the path consists of one Prefix Segment. The Prefix Segment steers the packets on a loop-free path from Node8 to Node10. The shortest path from Node8 to Node10 does not traverse a link of the protected SRLG and is along the post-convergence path.
- The PLR Node2 pushes the segment list {Prefix-SID(Node8), Prefix-SID(Node10)} on the repaired packets to destination Node6 and forwards them on interface to Node3.

The above behavior is applied per-prefix. The post-convergence path is computed for each destination prefix, and the TI-LFA repair path for each destination is then tailored over the post-convergence path to that destination. Note that the above description is a conceptual description of the computations instead of the actual algorithm. The actual TI-LFA algorithm is proprietary (local behavior which is not in the scope of IETF standardization) and scales extremely well.

Example 9-31 shows the ISIS computed TI-LFA repair path for destination 1.1.1.6/32 on Node2 for the topology in Figure 9-15. The output shows that it is an SRLG protecting repair path (line 5). The segments of the repair path are as illustrated in Figure 9-15. The repair path contains two Prefix Segments: to Node8 (16008) and to Node10 (16010). Node2 also swaps or imposes the Prefix-SID label of the destination Node6 (16006). ISIS does not instantiate a SRTE Policy since the label stack size is smaller than four (this platform supports up to 3 labels).

Example 9-31: ISIS TI-LFA local SRLG protecting repair path

```
 1 RP/0/0/CPU0:xrvr-2#show isis fast-reroute 1.1.1.6/32 detail
 2
 3 L2 1.1.1.6/32 [10/115] medium priority
 4      via 99.2.6.6, GigabitEthernet0/0/0/2, xrvr-6, SRGB
Base: 16000, Weight: 0
 5          Backup path: TI-LFA (srlg), via 99.2.3.3,
GigabitEthernet0/0/0/0 xrvr-3, SRGB Base: 16000, Weight: 0
 6            P node: xrvr-8.00 [1.1.1.8], Label: 16008
 7            P node: xrvr-10.00 [1.1.1.10], Label: 16010
 8            Prefix label: 16006
 9        P: No, TM: 60, LC: No, NP: No, D: No, SRLG: No
10      src xrvr-6.00-00, 1.1.1.6, prefix-SID index 6, R:0 N:1
P:0 E:0 V:0 L:0
```

The OSPF computed TI-LFA repair path on Node2 for the same topology is shown in Example 9-32. The path is SRLG protecting (`SRLG Disjoint`) (see line 14) and uses the segment list as illustrated in Figure 9-15. The nodes on the repair path are all labeled as `P node` (lines 11-12), the same as for ISIS.

Example 9-32: OSPF TI-LFA local SRLG protecting repair path

```
 1 RP/0/0/CPU0:xrvr-2#show ospf routes 1.1.1.6/32 backup-path
 2
 3 Topology Table for ospf 1 with ID 1.1.1.2
 4
 5 Codes: O - Intra area, O IA - Inter area
 6        O E1 - External type 1, O E2 - External type 2
 7|        O N1 - NSSA external type 1, O N2 - NSSA external
type 2
 8|
 9| O    1.1.1.6/32, metric 11
10|        99.2.6.6, from 1.1.1.6, via GigabitEthernet0/0/0/2,
path-id 1
11|            Backup path: TI-LFA,
12|                   Repair-List: P node: 1.1.1.8
Label: 16008
13|                               P node: 1.1.1.10
Label: 16010
14|               99.2.3.3, from 1.1.1.6, via
GigabitEthernet0/0/0/0, protected bitmap 0000000000000001
15|               Attributes: Metric: 61, SRLG Disjoint
```

The cef entry for Node6's loopback prefix 1.1.1.6/32 on Node2 is shown in Example 9-33. On the primary path Node2 imposes no label (`ImplNull` on line 15) since the destination Node6 is directly connected (Penultimate Hop Popping, PHP). On the repair (backup) path Node2 imposes or swaps Prefix-SID label 16006, and imposes labels 16008 and 16010 (line 10). Node2 then steers packets on the repair path to Node3 (via 99.2.3.3, Gi0/0/0/0) (line 6).

Example 9-33: TI-LFA SRLG protecting repair path in CEF

```
 1| RP/0/0/CPU0:xrvr-2#show cef 1.1.1.6/32
 2| 1.1.1.6/32, version 1759, internal 0x1000001 0x81 (ptr
0xa14564f4) [1], 0x0 (0xa143b92c), 0xa28 (0xa1750184)
 3|  Updated Jul  4 10:44:56.242
 4|  local adjacency 99.2.6.6
 5|  Prefix Len 32, traffic index 0, precedence n/a, priority 1
 6|    via 99.2.3.3/32, GigabitEthernet0/0/0/0, 26 dependencies,
weight 0, class 0, backup (remote) [flags 0x8300]
 7|     path-idx 0 NHID 0x0 [0xa10f3bd4 0x0]
 8|     next hop 99.2.3.3/32, P-node 1.1.1.8, Q-node 1.1.1.10
 9|     local adjacency
10|      local label 16006      labels imposed {16008 16010
16006}
11|    via 99.2.6.6/32, GigabitEthernet0/0/0/2, 26 dependencies,
weight 0, class 0, protected [flags 0x400]
```

```
    path-idx 1 bkup-idx 0 NHID 0x0 [0xa0e699dc 0xa0e695c8]
    next hop 99.2.6.6/32
     local label 16006      labels imposed {ImplNull}
```

The MPLS forwarding entry for Prefix-SID label 16006 on Node2 is displayed in Example 9-34. The repair path is shown in lines 14 to 21. The outgoing label stack is {16008 16010 16006 } (ordered Top → Bottom) and outgoing interface is Gi0/0/0/0 via 99.2.3.3.

Example 9-34: TI-LFA node protecting repair path in MPLS

```
 1 RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 16006 detail
 2 Local  Outgoing    Prefix             Outgoing     Next
Hop         Bytes
 3 Label  Label       or ID
Interface                    Switched
 4 ------ ----------- ------------------ ------------ ---------
------ ------------
 5 16006  Pop         SR Pfx (idx 6)     Gi0/0/0/2
99.2.6.6        0
 6      Updated: Jul  4 10:32:35.622
 7      Path Flags: 0x400 [  BKUP-IDX:0 (0xa0e699dc) ]
 8      Version: 1759, Priority: 1
 9      Label Stack (Top -> Bottom): { Imp-Null }
10      NHID: 0x0, Encap-ID: N/A, Path idx: 1, Backup path idx:
0, Weight: 0
11      MAC/Encaps: 14/14, MTU: 1500
12      Packets Switched: 0
13
14        16008       SR Pfx (idx 6)     Gi0/0/0/0
99.2.3.3        0            (!)
15      Updated: Jul  4 10:32:35.622
16      Path Flags: 0x8300 [  IDX:0 BKUP, NoFwd ]
17      Version: 1759, Priority: 1
18      Label Stack (Top -> Bottom): { 16008 16010 16006 }
19      NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx:
0, Weight: 0
20      MAC/Encaps: 14/26, MTU: 1500
21      Packets Switched: 0
22      (!): FRR pure backup
23
```

```
24|    Traffic-Matrix Packets/Bytes Switched: 0/0
```

9.5.3 De Facto Node Protection

TI-LFA provides a guaranteed node protection in any topology. However, a guaranteed node protection may not always be the best option. Given that link failures occur more often than node failures, it may not be a good idea to use the node protecting repair path for each failure. Remember that the protecting node does not differentiate between types of failures; only one repair path is installed for each destination and that is the one used for *any* failure. If a node protecting repair path is installed and only the link fails while the node stays up, still the node protecting repair path is used. In such case the repair path does not always follow the actual post-convergence path since the actual failure differs from the failure that was anticipated in the repair path calculation. This is true unless that the link protecting repair path also protects against node failures. For example in Figure 9-16, the link protecting repair path on PLR Node1 for destination Node3 is also protecting against failure of Node2 since the repair path does not traverse Node2.

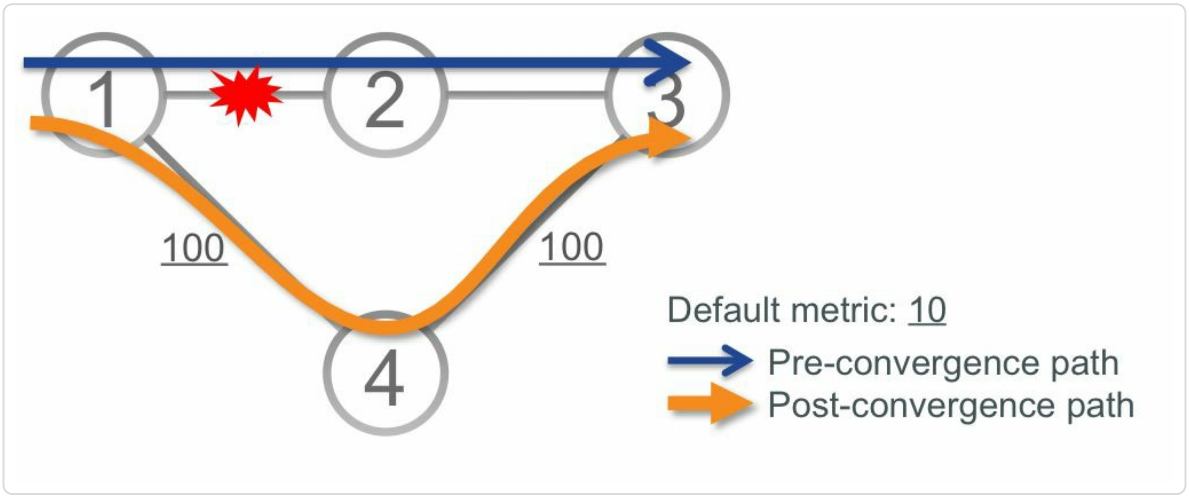


Figure 9-16: Link protecting repair path is also node protecting

Simulations indicate that non-guaranteed node-protecting repair paths actually provide *De Facto* ("by [the] fact") Node Protection in many cases. This means that the simultaneous link protection of different nodes can provide protection for a node failure. An example may clarify more. All nodes in the topology of Figure 9-17 have link protection enabled. The default IGP link metric is 30. The illustration shows various individual link failures on different nodes and the repair path that the affected node provides towards destination Node3. The failure of link Node1-Node2 in Figure 9-17 (a) triggers link protection on Node1. Link Node4-Node2 failure in Figure 9-17 (b) triggers link protection on Node4. And link failure Node5-Node2 in Figure 9-17 (c) triggers link protection on Node5. The individual link protecting repair paths in (a) and (b) are not node protecting since they all traverse Node2. Given that by failure of Node2, all the links of Node2 fail together, each link failure triggers the link protection on the affected node and the result is *de facto* node protection, as seen in Figure 9-17 (d). Node1's repair path for traffic destined for Node3 goes towards Node4. When the traffic arrives on Node4, Node4 steers the traffic to Node5 since Node4's repair path for destination Node3 goes towards Node5. When the traffic destined for Node3 arrives on Node5, Node5 steers the traffic to Node3 since Node5's repair path for destination Node3 goes directly to Node3.

The same reasoning can be applied to SRLG protection as well.

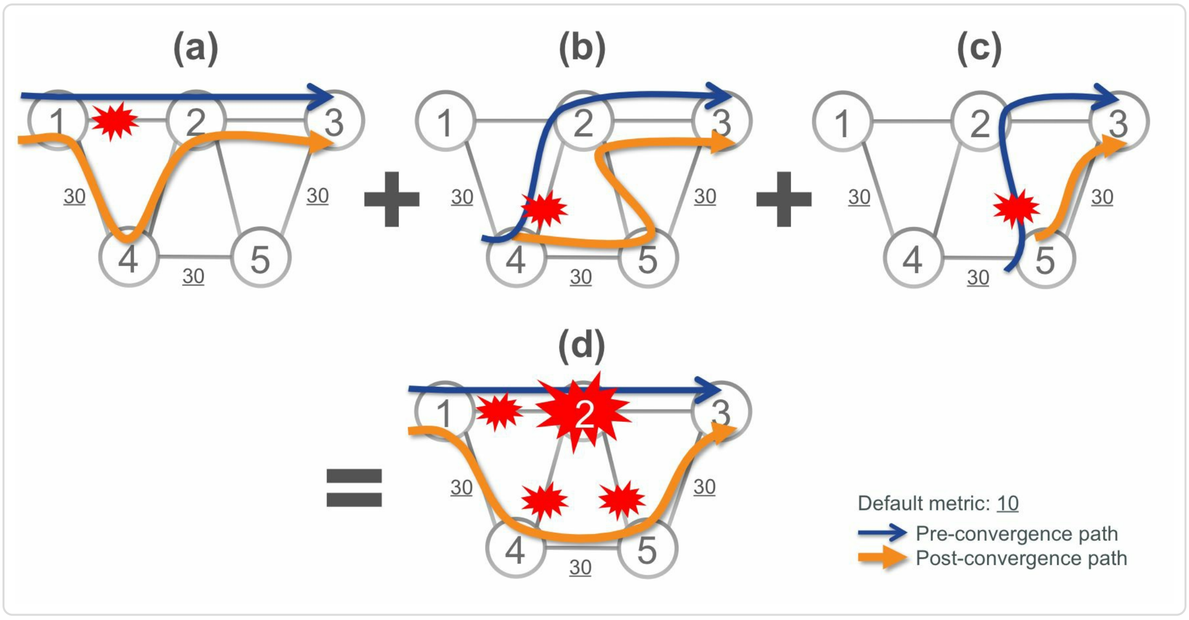


Figure 9-17: De facto node protection

De facto protection is topology dependent. A topological study or IPFRR simulation is required to find out if *de facto* protection is applicable. Leveraging *de facto* protection benefits from the optimal repair path for the more frequent link failures, while still providing protection for the less frequent node or SRLG failures.

"Protection for all types of services in the network in any topology via TI-LFA has been one of the prime and first use-case that has been driving SR deployment with operators. The key factor has been the operational simplicity and its automated nature – a single command to enable the feature on the desired interfaces is sufficient in many cases. The other most useful aspect has been the predictability of the backup path (via post convergence path) which has made it possible to perform capacity planning with failure simulations using tools like Cisco WAE. The quality of experience is also improved as there is no more switching of traffic back and forth – from original path to a repair path and then on to the post-convergence path. The backup being post-convergence path, which is determined by the IGP metrics used, makes it possible to avoid congestion on most failures with proper capacity planning."

— Ketan Talaulikar

9.6 Addressing P and PQ Nodes

In order to use a remote node as an intermediate hop (P-node or PQ-node) on the TI-LFA repair path, the protecting node needs to derive a Prefix-SID for that node from the link-state database. Similarly, if using Remote LFA, then the protecting node needs to find out the IP address of the PQ-node to establish the targeted LDP session. Logically the protecting node could pick any reachable node prefix (prefix with N-flag set) that is originated by the P-node and use the Prefix-SID of that prefix. Instead, pre-determined addresses are used by the Cisco IOS-XR implementation, as explained in this section. Since this is a local behavior choice on the PLR node, it is implementation specific.

HIGHLIGHT

It is strongly recommended to explicitly configure the "router-id" on each IGP node as one of the host loopback addresses that are also advertised by the IGP and provisioned with Prefix-SID; this becomes the default and preferred Node-SID for TI-LFA and removes ambiguities and ensures consistent repair paths

9.6.1 Addressing P and PQ Nodes in ISIS

ISIS selects the first host prefix it finds from the following set, in order of preference:

1. ISIS router-id (TLV 134) This address is configured as ISIS router-id or TE router-id
2. ISIS IP interface address (TLV 132) which advertises the IP addresses of one or more interfaces of this node. Cisco IOS XR advertises a

single interface address TLV per address family. Other vendors may advertise multiple of these. The address in this TLV is automatically selected by the advertising node as the address of the lowest numbered loopback interface advertised in ISIS, usually Loopback0, or the numerically lowest non-loopback interface address advertised in ISIS if no loopback interfaces are advertised.

3. The numerically highest host prefix advertised by that node; this really is a fallback option since the protecting node cannot always know if this prefix is local to the advertising node

If the selected prefix from this list does not have a Prefix-SID, then ISIS does not consider another address from the above list, but the node is assumed to not have a reachable Prefix-SID. Therefore it is important to ensure that a Prefix-SID is configured for the most preferred address from the above list. If the TE Router-id is configured, then the TE Router-id prefix must have an associated Prefix-SID. If no TE Router-id is configured, then the prefix of the lowest numbered loopback interface must have an associated Prefix-SID. If multiple loopbacks are configured on the node and the lowest numbered loopback interface does not have a Prefix-SID configured, then the loopback prefix that has an associated Prefix-SID must be configured as TE Router-id.

As an example, a node has the configuration shown in Example 9-35. Notice that no router-id is configured and Loopback0 has no Prefix-SID configured. Based on this configuration and the preference order above, a remote node would select 1.1.1.1 to reach this node since this is the address in the IP interface address TLV (named `IP address` in the output). See line 12 of the ISIS advertisement in Example 9-36. However, since the prefix 1.1.1.1/32 has no associated Prefix-SID, the protecting node assumes this node does not have a Prefix-SID.

Example 9-35: ISIS IP interface address TLV – configuration

```
interface Loopback0
 ipv4 address 1.1.1.1 255.255.255.255
 ipv6 address 2001::1:1:1:1/128
!
interface Loopback1
 ipv4 address 1.0.0.1 255.255.255.255
 ipv6 address 2001::1:0:0:1/128
!
router isis 1
 is-type level-2-only
 net 49.0001.0000.0000.0001.00
 address-family ipv4 unicast
  metric-style wide
  segment-routing mpls
 !
 address-family ipv6 unicast
  metric-style wide
  segment-routing mpls
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
 !
 interface Loopback1
  passive
  address-family ipv4 unicast
   prefix-sid absolute 16001
  !
  address-family ipv6 unicast
   prefix-sid absolute 20001
  !
 !
```

Example 9-36: ISIS IP interface address TLV – advertisement

```
RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-1

IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
xrvr-1.00-00        * 0x00000003   0x01a2        1144            0/0/0
  Area Address:   49.0001
  NLPID:          0xcc
  NLPID:          0x8e
  MT:             Standard (IPv4 Unicast)
  MT:             IPv6 Unicast                                     0/0/0
  Hostname:       xrvr-1
  IP Address:     1.1.1.1
  IPv6 Address:   2001::1:1:1:1
  Router Cap:     1.1.1.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 0          IP-Extended 1.0.0.1/32
    Prefix-SID Index: 101, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 0          IP-Extended 1.1.1.1/32
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:0:0:1/128
    Prefix-SID Index: 4101, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128

 Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

To remedy this, the operator can configure a router-id. Configuring a TE router-id also requires enabling the TE infrastructure and advertising the TE link attributes. Another option is to configure the router-id as ISIS router-id. Both configuration options result in advertisement of TLV 134, but the latter does not require advertising the TE link attributes. The configuration is shown in Example 9-37 and the resulting ISIS advertisement in Example 9-38. With this configuration the protecting node selects prefix 1.0.0.1/32 that has an associated Prefix-SID.

Example 9-37: ISIS router-id – configuration

```
interface Loopback0
 ipv4 address 1.1.1.1 255.255.255.255
 ipv6 address 2001::1:1:1:1/128
!
interface Loopback1
 ipv4 address 1.0.0.1 255.255.255.255
 ipv6 address 2001::1:0:0:1/128
!
router isis 1
 address-family ipv4 unicast
  router-id Loopback1
 address-family ipv6 unicast
  router-id Loopback1
```

Example 9-38: ISIS router-id – advertisement

```
RP/0/0/CPU0:xrvr-1#show isis database verbose xrvr-1

IS-IS 1 (Level-2) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime
ATT/P/OL
xrvr-1.00-00        * 0x00000009   0x54fe
1195            0/0/0
  Area Address:   49.0001
  NLPID:          0xcc
  NLPID:          0x8e
  MT:             Standard (IPv4 Unicast)
  MT:             IPv6 Unicast
0/0/0
  Hostname:       xrvr-1
  IP Address:     1.1.1.1
  IPv6 Address:   2001::1:1:1:1
  Router ID:      1.0.0.1
  IPv6 Router ID: 2001::1:0:0:1
  Router Cap:     1.0.0.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
  Metric: 0          IP-Extended 1.0.0.1/32
    Prefix-SID Index: 101, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Metric: 0          IP-Extended 1.1.1.1/32
  Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:0:0:1/128
    Prefix-SID Index: 4101, Algorithm:0, R:0 N:1 P:0 E:0 V:0
L:0
```

```
    Metric: 0          MT (IPv6 Unicast) IPv6 2001::1:1:1:1/128

   Total Level-2 LSP count: 1     Local Level-2 LSP count: 1
```

9.6.2 Addressing P and PQ Nodes in OSPF

To reach a TI-LFA P-node (or PQ-node) from the protecting node, OSPF prefers the OSPF router-id of the P-node if it is a reachable prefix. Otherwise OSPF selects the numerically highest reachable host address with the N-flag set that is advertised by the P-node. If the selected prefix has no Prefix-SID then the protecting node assumes that the P-node does not advertise a Prefix-SID..

9.7 Protecting Adjacency Segments

TI-LFA protects both Prefix Segments and Adjacency Segments. To protect an Adjacency Segment, TI-LFA steers the packets on a repair path to the node on the remote end of the link, the adjacent node.[4] From there the packets resume their normal path. The following logic is used to protect an Adjacency Segment:

1. If the link of the Adjacency Segment is on the shortest path to the neighbor of the adjacency, then the repair path to protect the Adjacency Segment is equal to the repair path of the neighbor's Node-SID.
2. If the link of the Adjacency Segment is not on the shortest path to the neighbor of the adjacency, then the repair path to protect the Adjacency Segment is equal to the primary path to the neighbor's Node-SID.

Any tiebreakers that are configured to influence the repair path to the neighbor's Node-SID then also influence the repair path for the Adjacency-SID, since the same repair path is used. If the repair path of the neighbor's Node-SID requires an SRTE Policy (since the number of labels is larger than the native platform support), then also the Adjacency-SID's repair path uses that SRTE Policy.

A repair path is only installed for the protection-eligible Adjacency-SID. See the IGP Control plane chapter 5 for more details about protection-eligible and unprotected Adjacency-SIDs.

The IGP directly installs the Adjacency Segment MPLS forwarding entry in LSD without involving FIB. This includes the Adjacency Segment's repair path. This affects the possibility of SR/LDP interworking functionality on the Adjacency Segment repair path. FIB takes care of the SR/LDP interworking functionality, as discussed in chapter 7, "Segment Routing in Existing MPLS Networks". Since FIB is not involved in programming the Adjacency Segment's repair path, this repair path cannot benefit from SR/LDP interworking functionality. Section 9.8.3 discusses this in more detail.

Figure 9-18 illustrates the Adjacency Segment protection. All links in the topology have a metric 10, except for the link between Node2 and Node4 that has a metric 30. Based on these metrics the primary path from Node2 to Node1 is via the direct link between these two nodes. Therefore, based on the explanation above, the repair path of the Adjacency Segment Node2-Node1 follows the same path as the repair path of the neighbor's Prefix-SID: via PQ-node Node4. In the case the primary link between Node2 and Node1 fails, then the PLR Node2 swaps the top label (the Adjacency-SID label) with the Prefix-SID label of the node at the remote end of the link and pushes the repair path label stack on top. In the example the incoming Adjacency-SID label 30201 is swapped with Prefix-SID label 16001 (Prefix-SID of Node1) and the Prefix-SID label of the PQ-node Node4 is pushed on the packet and steered on the interface to Node3.

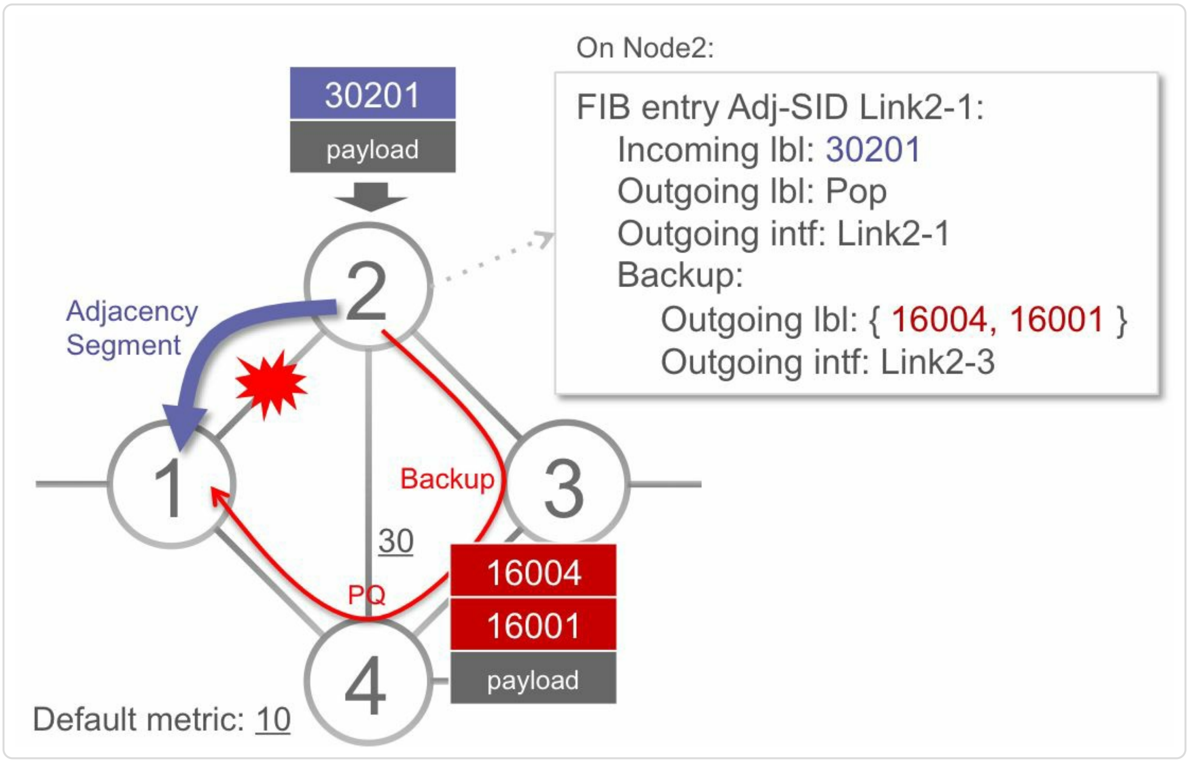


Figure 9-18: Adjacency Segment protection

The ISIS output on Node2 for this example is shown in Example 9-39 and Example 9-40. Example 9-39 shows the repair path on Node2 for the loopback prefix of Node1, 1.1.1.1/32. The repair path goes via PQ-node Node4, using Prefix-SID label 16004. Example 9-40 shows the repair path on Node2 for the Adjacency Segment of the adjacency to Node1. This link is on the shortest path from Node2 to Node1, therefore the repair path of this Adjacency Segment is equal to the repair path of 1.1.1.1/32. The repair path label stack (line 10) and outgoing interface (lines 12-13) are the same as for the repair path of destination 1.1.1.1/32.

Example 9-39: ISIS Adjacency-SID protection – remote node's Prefix-SID protection

```
1 RP/0/0/CPU0:xrvr-2#show isis fast-reroute 1.1.1.1/32
2
3 L2 1.1.1.1/32 [10/115]
4      via 99.1.2.1, GigabitEthernet0/0/0/0, xrvr-1, SRGB Base:
16000, Weight: 0
5          Backup path: TI-LFA (link), via 99.2.3.3,
GigabitEthernet0/0/0/1 xrvr-3, SRGB Base: 16000, Weight: 0
             P node: xrvr-4.00 [1.1.1.4], Label: 16004
             Prefix label: 16001
```

Example 9-40: ISIS Adjacency-SID protection

```
RP/0/0/CPU0:xrvr-2#show isis adjacency Gi0/0/0/2 detail

IS-IS 1 Level-2 adjacencies:
System Id      Interface        SNPA           State Hold
Changed  NSF IPv4 IPv6

 BFD  BFD
xrvr-1         Gi0/0/0/0        *PtoP*         Up    23
00:49:56 Yes None None
  Area Address:           49.0001
  Neighbor IPv4 Address:  99.1.2.2*
  Adjacency SID:          30201 (protected)
   Backup label stack:    [16004, 16001]
   Backup stack size:     2
   Backup interface:      Gi0/0/0/1
   Backup nexthop:        99.2.3.3
   Backup node address:   1.1.1.1
  Non-FRR Adjacency SID:  310201
  Topology:               IPv4 Unicast

Total adjacency count: 1
```

The OSPF output for this example is shown in Example 9-41, Example 9-42, and Example 9-43. Example 9-41 shows the OSPF TI-LFA repair path for the loopback prefix of Node1, 1.1.1.1/32. The repair path goes via PQ-node Node4. Line 22 of Example 9-42 shows that the Adjacency-SID of the adjacency to Node1 is protected. Line 18 of Example 9-43 shows the label stack of the repair path for the Adjacency-SID label.

Example 9-41: OSPF Adjacency-SID protection – remote node's Prefix-SID protection

```
RP/0/0/CPU0:xrvr-2#show ospf routes 1.1.1.1/32 backup-path

Topology Table for ospf 1 with ID 1.1.1.2
```

```
Codes: O - Intra area, O IA - Inter area
       O E1 - External type 1, O E2 - External type 2
       O N1 - NSSA external type 1, O N2 - NSSA external type 2

O    1.1.1.1/32, metric 2
       99.1.2.1, from 1.1.1.1, via GigabitEthernet0/0/0/0,
path-id 1
           Backup path: TI-LFA, Repair-List: P node:
1.1.1.4        Label: 16004
              99.2.3.3, from 1.1.1.1, via
GigabitEthernet0/0/0/1, protected bitmap 0000000000000001
              Attributes: Metric: 40, SRLG Disjoint
```

Example 9-42: OSPF Adjacency-SID protection – OSPF neighbor output

```
RP/0/0/CPU0:xrvr-2#show ospf neighbor detail 1.1.1.1

* Indicates MADJ interface
# Indicates Neighbor awaiting BFD session up

Neighbors for OSPF 1

 Neighbor 1.1.1.1, interface address 99.1.2.1
    In the area 0 via interface GigabitEthernet0/0/0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:00:37
    Neighbor is up for 00:14:07
    Number of DBD retrans during last exchange 0
    Index 3/3, retransmission queue length 0, number of
retransmission 1
    First 0(0)/0(0) Next 0(0)/0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
    LS Ack list: NSR-sync pending 0, high water mark 0
    Adjacency SID Label: 30201, Protected
    Unprotected Adjacency SID Label: 310201


Total neighbor count: 1
```

Example 9-43: OSPF Adjacency-SID protection – MPLS forwarding

```
RP/0/0/CPU0:xrvr-2#show mpls forwarding labels 30201 detail
Local  Outgoing    Prefix             Outgoing     Next
Hop        Bytes
Label  Label       or ID
       Interface                     Switched
------ ----------- ------------------ ------------ ------------
--- ------------
30201  Pop         SR Adj (idx 0)     Gi0/0/0/0
99.1.2.1        0
     Updated: Jun 30 16:21:26.553
     Path Flags: 0x400 [  BKUP-IDX:1 (0xa0e69ac4) ]
     Version: 201, Priority: 1
     Label Stack (Top -> Bottom): { Imp-Null }
     NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx: 1,
Weight: 1
     MAC/Encaps: 14/14, MTU: 1500
     Packets Switched: 0

       16004       SR Adj (idx 0)     Gi0/0/0/1
99.2.3.3        0            (!)
     Updated: Jun 30 16:21:26.553
     Path Flags: 0x100 [  BKUP, NoFwd ]
     Version: 201, Priority: 1
     Label Stack (Top -> Bottom): { 16004 16001 }
     NHID: 0x0, Encap-ID: N/A, Path idx: 1, Backup path idx: 0,
Weight: 4
     MAC/Encaps: 14/22, MTU: 1500
     Packets Switched: 0
     (!): FRR pure backup
```

9.8 TI-LFA for IP and LDP Traffic

TI-LFA protection is not limited to SR traffic, but can also be leveraged by LDP traffic and unlabeled IP (IPv4 and IPv6) traffic. In steady state, an operator can keep the unlabeled IP traffic as unlabeled traffic and keep traffic that is carried over LDP LSPs as LDP carried traffic but in the case of failures, they can benefit from the enhancements that Segment Routing and TI-LFA brings for protection. SR transport is then only used on the repair path in case of failure. The traffic is carried by its original transport again after IGP converges.

To protect LDP carried traffic, the destination prefix of this traffic must have a Prefix-SID. If the node that originates the destination prefix cannot advertise a Prefix-SID for that prefix (it is for example an LDP-only node), then a Mapping Server must advertise a Prefix-SID for this prefix. TI-LFA then uses this Prefix-SID (derived from SRMS) for the repair path of this prefix. For this reason, it is required that all the SR nodes on the path from the TI-LFA release point to the destination node support the Mapping Client functionality. This functionality is needed to forward the traffic using the Prefix-SID advertised by the Mapping Server.

To protect unlabeled IP traffic, no Prefix-SID is required for the destination prefix of the unlabeled traffic. Before the failure the traffic is kept unlabeled. In the case of failure, the label stack for the repair path is imposed on the packets and the packets are released as unlabeled IP packets at the release point at the end of the repair path. When IGP converges after the failure then the traffic is again transported without label. Somewhat outside of the discussion: if the traffic needs to stay unlabeled then also LDP label imposition must be prevented. If LDP is enabled then by default Cisco IOS XR imposes LDP labels on unlabeled incoming packets. Filtering LDP label allocation or label advertisement can prevent this.

To protect unlabeled IP traffic on a Cisco IOS XR platform SR node, Segment Routing must be enabled in the global configuration on the protecting node, as shown in Example 9-44. By applying this configuration, local labels are allocated for the unlabeled (non-SR) destinations that are protected by TI-LFA. These local labels are required for the two-stage forwarding mechanism that is used in most Cisco IOS XR platforms. Without a local label no outgoing label can be imposed on the protected packets.

Example 9-44: Enable Segment Routing in global configuration

```
segment-routing
!
```

REMINDER: Two-stage forwarding

To get better scaling and performance, most Cisco IOS XR platforms use a two-stage forwarding data plane architecture.

In the first stage, the ingress line card performs a FIB lookup for the destination address of an incoming packet. The FIB lookup returns enough information to deliver the packet to the egress line card. The ingress line card then sends the packet through the fabric to the egress line card.

In the second stage, the egress line card performs a FIB lookup on the packet's destination address. The FIB lookup returns the full adjacency and Layer-2 rewrite information. The egress line card then sends the packet to the outgoing interface.

For this two-stage forwarding to work for labeled packets, a local label is required to associate the forwarding entry on the ingress line card to the forwarding entry on the egress line card. The ingress line card imposes the local label on the ingress packet and sends the packet to the egress line card, which then swaps the local label with the correct outgoing label and sends the packets out.

If LDP is enabled on the protecting node then LDP would allocate a local label for the destination prefixes of the unlabeled IP traffic.

TI-LFA protection uses the SR/LDP interworking functionality for the repair paths where possible. Although it is strongly recommended to enable SR on all nodes that the TI-LFA repair path traverses, not all nodes on the TI-LFA repair path *must* be Segment Routing capable. This is explained in more detail in section 9.8.3, "TI-LFA and SR/LDP Interworking".

9.8.1 Protecting Unlabeled IP Traffic

The network topology in Figure 9-19 is the same as in Figure 9-12. All nodes in the topology are SR enabled, except the source and destination nodes (Node11 and Node6) that are non-MPLS nodes. LDP is not enabled on any node. The loopback prefix 1.1.1.6/32 of Node6 does not have an associated Prefix-SID.

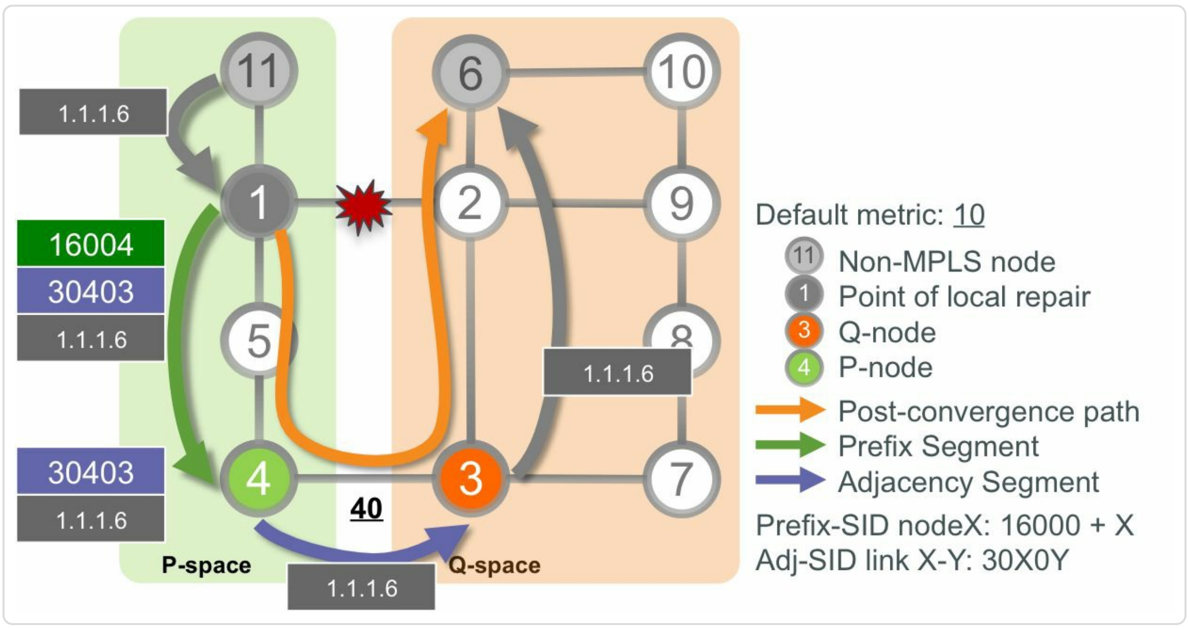


Figure 9-19: TI-LFA link protection of unlabeled IP traffic

To protect traffic to destination Node6 on PLR Node1, TI-LFA computed a double-segment repair path via P-node Node4 and Q-node Node3. This repair path also protects IP traffic. In the case that link Node1-Node2 fails, the PLR Node1 imposes the repair path segments on the protected packets and steers them towards Node4. The packets on the repair path are released on release point Node3. From Node3 the traffic continues as unlabeled IP traffic to its destination.

Example 9-45 shows the RIB entry on Node1 for destination prefix 1.1.1.6/32 of Node6.

The primary path for this prefix is shown in lines 17 to 26 of the output. IGP has installed the RIB entry for this prefix. Since the prefix has no associated Prefix-SID, IGP did not provide a local label for that prefix to RIB. Therefore the RIB entry shows `No local label` on line 28 of the output. Since the prefix has no associated Prefix-SID, also the primary path has no outgoing label (line 19).

The repair path for this prefix is shown in lines 7 to 15 of the output. The outgoing label stack consists of two labels (line 10), from top to bottom (left to right): Prefix-SID of P-node Node4 (16004), and Adjacency-SID of link Node4-Node3 (30403).

Example 9-45: RIB entry of unlabeled IP prefix

```
 1| RP/0/0/CPU0:xrvr-1#show route 1.1.1.6/32 detail
 2|
 3| Routing entry for 1.1.1.6/32
 4|   Known via "ospf 1", distance 110, metric 21, type intra
area
 5|   Installed Jul  5 13:18:38.580 for 00:01:48
 6|   Routing Descriptor Blocks
 7|     99.1.5.5, from 1.1.1.6, via GigabitEthernet0/0/0/0,
Backup (remote)
```

```
 8|       Remote LFA is 1.1.1.4, 1.1.1.3
 9|       Route metric is 0
10|       Labels: 0x3e84 0x76c3 (16004 30403)
11|       Tunnel ID: None
12|       Binding Label: None
13|       Extended communities count: 0
14|       Path id:65               Path ref count:1
15|       NHID:0x2(Ref:48)
16|       OSPF area:
17|     99.1.2.2, from 1.1.1.6, via GigabitEthernet0/0/0/1,
Protected
18|       Route metric is 21
19|       Label: None
20|       Tunnel ID: None
21|       Binding Label: None
22|       Extended communities count: 0
23|       Path id:1       Path ref count:0
24|       NHID:0x4(Ref:44)
25|       Backup path id:65
26|       OSPF area: 0
27|   Route version is 0x7 (7)
28|   No local label
29|   IP Precedence: Not Set
30|   QoS Group ID: Not Set
31|   Flow-tag: Not Set
32|   Fwd-class: Not Set
33|   Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD
Type RIB_SVD_TYPE_LOCAL
34|   Download Priority 1, Download Version 3013
35|   No advertising protos.
```

Example 9-46 shows the CEF entry for prefix 1.1.1.6/32. The primary path of the prefix is shown in lines 11 to 14 of the output. A local label 90106 has been allocated for this prefix, as shown in line 14. No label is imposed on the packets forwarded on the primary path; the output shows `labels imposed {None}` on line 14. The repair path is shown in lines 6 to 10. The imposed label stack is {16004, 30403} (line 10). This is the same label stack as the one of the RIB output.

Example 9-46: CEF entry of unlabeled IP prefix

```
RP/0/0/CPU0:xrvr-1#show cef 1.1.1.6/32
1.1.1.6/32, version 1155, internal 0x1000001 0x5 (ptr 0xa1435274) [1], 0x0 (0xa1400784), 0xa28 (0xa171c7b4)
 Updated May  4 09:08:31.589
 local adjacency 99.1.2.2
 Prefix Len 32, traffic index 0, precedence n/a, priority 15
   via 99.1.5.5/32, GigabitEthernet0/0/0/0, 11 dependencies, weight 0, class 0, backup [flags 0x300]
    path-idx 0 NHID 0x0 [0xa110d250 0x0]
    next hop 99.1.5.5/32, P-node 1.1.1.4, Q-node 1.1.1.3
    local adjacency
     local label 90106      labels imposed {16004 30403}
   via 99.1.2.2/32, GigabitEthernet0/0/0/1, 11 dependencies, weight 0, class 0, protected [flags 0x400]
    path-idx 1 bkup-idx 0 NHID 0x0 [0xa0e843d4 0x0]
    next hop 99.1.2.2/32
     local label 90106      labels imposed {None}
```

The protection of unlabeled IP traffic also applies for the case where an SRTE Policy is instantiated to support a larger label stack on the repair path. Figure 9-20 repeats Figure 9-14 where node protection is enabled on the PLR Node1. Four labels are required on the repair path; hence an SRTE Policy is instantiated on this platform.

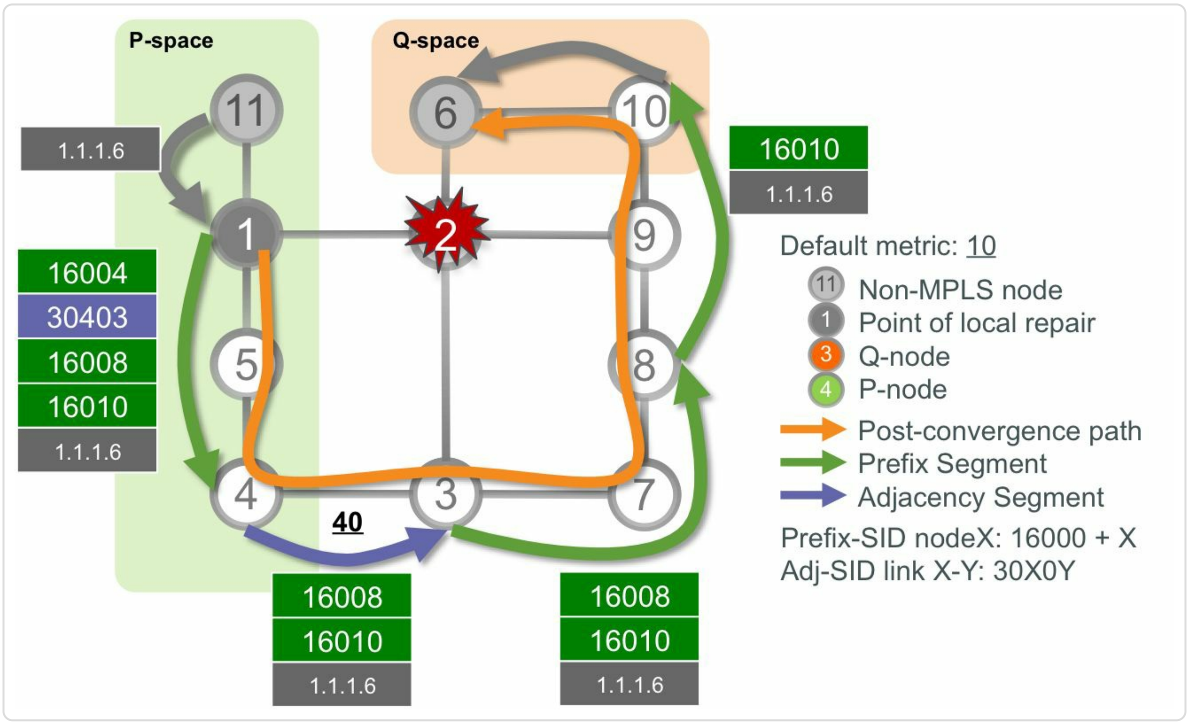


Figure 9-20: TI-LFA node protection of unlabeled IP traffic

The RIB entry of prefix 1.1.1.6/32 on Node1 is shown in Example 9-47. The repair path is shown in lines 17 to 25 of the output. The outgoing label 0x100004 (1048580) in line 19 is an internal value used to represent “pop”. This means that no additional label is imposed on the packet before it is steered into the SRTE Policy. Therefore the packet continues its travel from the release point to its destination as an unlabeled packet.

Example 9-47: RIB entry of an unlabeled IP prefix using SRTE Policy

```
 1 RP/0/0/CPU0:xrvr-1#show route 1.1.1.6/32 detail
 2
 3 Routing entry for 1.1.1.6/32
 4   Known via "ospf 1", distance 110, metric 21, type intra
area
 5   Installed Jul  6 14:39:56.727 for 00:09:06
 6   Routing Descriptor Blocks
 7     99.1.2.2, from 1.1.1.6, via GigabitEthernet0/0/0/1,
Protected
 8       Route metric is 21
 9       Label: None
```

```
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        Path id:1       Path ref count:0
        NHID:0x4(Ref:42)
        Backup path id:66
        OSPF area: 0
      directly connected, via tunnel-te32800, Backup
        Route metric is 0
        Label: 0x100004 (1048580)
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        Path id:66              Path ref count:1
        NHID:0x20(Ref:2)
        OSPF area:
    Route version is 0xc1 (193)
    No local label
    IP Precedence: Not Set
    QoS Group ID: Not Set
    Flow-tag: Not Set
    Fwd-class: Not Set
    Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD Type RIB_SVD_TYPE_LOCAL
    Download Priority 1, Download Version 3294
    No advertising protos.
```

The CEF entry for prefix 1.1.1.6/32 on Node1 is shown in Example 9-48. No label is imposed on the primary path (`labels imposed {None}` on line 9). The traffic on the repair is steered into the SRTE Policy tunnel-te32800 (line 10) without imposing an additional label on top of the repair path's label stack (`labels imposed {ImplNull}` in line 14).

Example 9-48: CEF entry of unlabeled IP prefix using SRTE Policy

```
1 RP/0/0/CPU0:xrvr-1#show cef 1.1.1.6/32
2 1.1.1.6/32, version 1511, internal 0x1000001 0x5 (ptr 0xa1434d74) [1], 0x0 (0xa13ff680), 0xa20 (0xa171c208)
3  Updated May  4 09:08:31.589
4  local adjacency 99.1.2.2
5  Prefix Len 32, traffic index 0, precedence n/a, priority 15
```

```
    via 99.1.2.2/32, GigabitEthernet0/0/0/1, 5 dependencies,
weight 0, class 0, protected [flags 0x400]
     path-idx 0 bkup-idx 1 NHID 0x0 [0xa0e83e64 0x0]
     next hop 99.1.2.2/32
      local label 90106      labels imposed {None}
    via 0.0.0.0/32, tunnel-te32800, 5 dependencies, weight 0,
class 0, backup [flags 0x300]
     path-idx 1 NHID 0x0 [0xa110d9dc 0xa110d934]
     next hop 0.0.0.0/32
     local adjacency
      local label 90106      labels imposed {ImplNull}
```

9.8.2 Protecting LDP Traffic

The network topology in Figure 9-21 is the same as in Figure 9-12. The difference is that all nodes in this topology have both SR and LDP enabled. Except for Node11 and Node6, which are LDP-only nodes. PLR Node1 computes a link protecting repair path for destination Node6. A Prefix-SID for Node6 is required in order to transport the protected packets as labeled packets from the repair path's release point Node3 to the destination Node6. Since Node6 is a non-SR node, a Mapping Server must advertise that Prefix-SID for Node6.

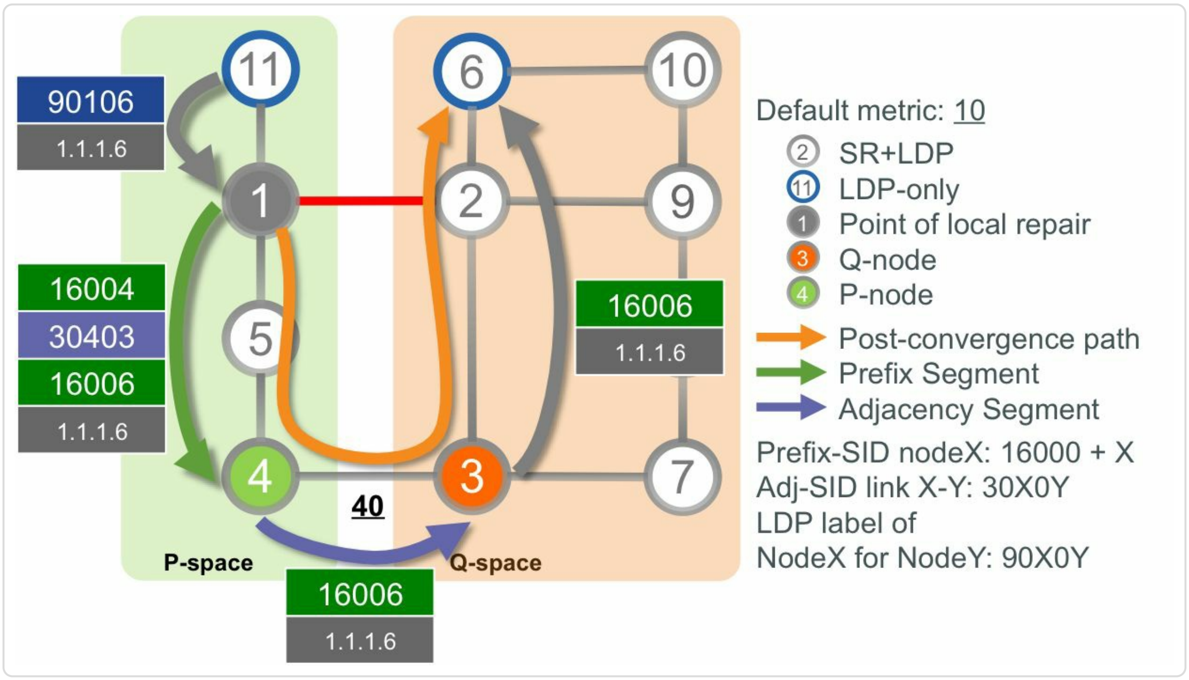


Figure 9-21: TI-LFA protecting LDP traffic

The Active Mapping Policy displayed in Example 9-49 indicates that PLR Node1 has learned the prefix-to-SID mapping for Node6's loopback prefix 1.1.1.6/32. The Prefix-SID for prefix 1.1.1.6/32 has a SID index 6. Since all nodes use the default SRGB [16000-23999], the Prefix-SID label is 16006 (= 16000 + 6).

Example 9-49: ISIS Active Mapping Policy on Node1

```
RP/0/0/CPU0:xrvr-1#show isis segment-routing prefix-sid-map
active-policy
IS-IS 1 active policy
Prefix               SID Index    Range        Flags
1.1.1.6/32           6            1

Number of mapping entries: 1
```

ISIS on Node1 has computed a TI-LFA repair path for destination prefix 1.1.1.6/32. The repair path goes via P-node Node4 and Q-node Node3, as shown in Example 9-50. The label 30403 (line 7) to go from P-node to Q-node is the Adjacency-SID label of the adjacency from Node4 to Node3. The Prefix-SID label 16006 (line 8) is as advertised by the Mapping Server for prefix 1.1.1.6/32.

Example 9-50: ISIS link protecting repair path

```
1 RP/0/0/CPU0:xrvr-1#show isis fast-reroute 1.1.1.6/32
2
3 L2 1.1.1.6/32 [20/115]
4      via 99.1.2.2, GigabitEthernet0/0/0/1, xrvr-2, SRGB Base:
16000, Weight: 0
5          Backup path: TI-LFA (link), via 99.1.5.5,
GigabitEthernet0/0/0/0 xrvr-5, SRGB Base: 16000, Weight: 0
6            P node: xrvr-4.00 [1.1.1.4], Label: 16004
7            Q node: xrvr-3.00 [1.1.1.3], Label: 30403
8            Prefix label: 16006
```

ISIS installs the prefix entry in RIB, as shown in Example 9-51. The primary path is displayed in lines 16 to 24 and the repair path in lines 7 to 15. The local label for this prefix is the Prefix-SID label 16006 (line 26). On the primary path, the outgoing label is 16006 (line 18) and on the repair path, the outgoing label stack is {16004, 30403, 16006}. Notice that all the labels in this RIB entry are all Segment Routing labels because in this example, all nodes are SR-enabled and advertise their SIDs, except Node11 and Node6. A Mapping Server (SRMS) advertises a Prefix-SID 16006 for prefix 1.1.1.6/32 (Node6). This is also indicated in the RIB entry by flagging the entry as `labeled SR(SRMS)` (line 4).

Example 9-51: ISIS RIB entry

```
1 RP/0/0/CPU0:xrvr-1#show route 1.1.1.6/32 detail
2
3 Routing entry for 1.1.1.6/32
4   Known via "isis 1", distance 115, metric 20, labeled
SR(SRMS), type level-2
5   Installed Jul  8 15:54:15.394 for 00:02:14
6   Routing Descriptor Blocks
```

```
      99.1.5.5, from 1.1.1.6, via GigabitEthernet0/0/0/0, Backup (remote)
        Remote LFA is 1.1.1.4, 1.1.1.3
        Route metric is 0
        Labels: 0x3e84 0x76c3 0x3e86 (16004 30403 16006)
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        Path id:65             Path ref count:1
        NHID:0x2(Ref:24)
      99.1.2.2, from 1.1.1.6, via GigabitEthernet0/0/0/1, Protected
        Route metric is 20
        Label: 0x3e86 (16006)
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        Path id:1       Path ref count:0
        NHID:0x4(Ref:20)
        Backup path id:65
    Route version is 0xcd (205)
    Local Label: 0x3e86 (16006)
    IP Precedence: Not Set
    QoS Group ID: Not Set
    Flow-tag: Not Set
    Fwd-class: Not Set
    Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD Type RIB_SVD_TYPE_LOCAL
    Download Priority 1, Download Version 3451
    No advertising protos.
```

If using OSPF in this topology, Node1 received the Mapping Server advertisement for prefix 1.1.1.6/32 via OSPF. See Example 9-52. OSPF computed the same TI-LFA repair path as ISIS, as shown in Example 9-53.

Example 9-52: OSPF Active Mapping Policy on Node1

```
RP/0/0/CPU0:xrvr-1#show ospf segment-routing prefix-sid-map active-policy
```

```
        SRMS active policy for Process ID 1

Prefix                    SID Index     Range         Flags
1.1.1.6/32                6             1

Number of mapping entries: 1
```

Example 9-53: OSPF link protecting repair path

```
RP/0/0/CPU0:xrvr-1#show ospf route 1.1.1.6/32 backup-path
detail

OSPF Route entry for 1.1.1.6/32
  Route type:  Intra-area
  Last updated: Jul  8 15:04:41.398
  Metric: 21
    SPF priority: 4,  SPF version: 117
  RIB version: 0,  Source: Unknown
       99.1.2.2, from 1.1.1.6, via GigabitEthernet0/0/0/1,
path-id 1
           Backup path: TI-LFA, Repair-List:
                              P node: 1.1.1.4         Label:
16004
                              Q node: 1.1.1.3         Label:
30403
              99.1.5.5, from 1.1.1.6, via
GigabitEthernet0/0/0/0, protected bitmap 0000000000000001
              Attributes: Metric: 81, SRLG Disjoint
```

The RIB and CEF entries based on the OSPF information are equivalent to the ones seen with ISIS in the network.

All nodes in this network also have LDP enabled. PLR Node1 has an LDP session with its three neighbors Node2, Node5, and Node11. Node1 allocates a local LDP label 90106 for destination Node6's loopback prefix 1.1.1.6/32 and learns the LDP label bindings for this prefix from its LDP neighbors (Node2, Node5, and Node11). All these LDP labels are shown in Example 9-54.

Example 9-54: LDP label bindings on PLR

```
RP/0/0/CPU0:xrvr-1#show mpls ldp bindings 1.1.1.6/32
1.1.1.6/32, rev 959
        Local binding: label: 90106
        Remote bindings: (3 peers)
            Peer                Label
            -----------------   ---------
            1.1.1.2:0           90206
            1.1.1.5:0           90506
            1.1.1.11:0          91106
```

Node1 also installs the LDP forwarding entry for prefix 1.1.1.6/32; see Example 9-55. The local LDP label 90106 for 1.1.1.6/32 is shown on line 11 in the second column of the output, labeled `Label In`. The primary path to prefix 1.1.1.6/32 is shown in the last line of the output. The outgoing LDP label for this primary path is 90206, which is the label that the downstream neighbor Node2 has allocated for prefix 1.1.1.6/32. The outgoing interface and next-hop of the primary path are towards Node2.

The repair path for prefix 1.1.1.6/32 is shown in lines 11 to 13 of the output. The top label 90504 of the outgoing label stack (line 11) is the LDP outgoing label to the P-node Node4. This outgoing LDP label is advertised by the downstream LDP neighbor Node5 for Node4's prefix 1.1.1.4/32. See the LDP label bindings for prefix 1.1.1.4/32 on Node1 as displayed in Example 9-56. The LDP label 90504 that Node5 advertised for prefix 1.1.1.4/32 is displayed on line 8 of this output. The last two labels in the three-label stack are `Unlabelled` (lines 12-13). These are the label entries to steer the packets on the repair path from P-node Node4 to Q-node Node3 and from there to destination Node6. LDP does not have label values for these entries. CEF uses the SR/LDP interworking functionality to replace these unlabeled entries with valid label values.

Example 9-55: LDP forwarding entry

```
RP/0/0/CPU0:xrvr-1#show mpls ldp forwarding 1.1.1.6/32

Codes:
  - = GR label recovering, (!) = LFA FRR pure backup path
  {} = Label stack with multi-line output for a routing path
  G = GR, S = Stale, R = Remote LFA FRR backup

Prefix          Label   Label(s)       Outgoing     Next
Hop             Flags
                In      Out
Interface                     G S R
--------------- ------- -------------- ------------ --------
----------- -----
1.1.1.6/32      90106   { 90504        Gi0/0/0/0
99.1.5.5        (!)     R
                          Unlabelled
(1.1.1.4)
                          Unlabelled }
(1.1.1.3)
                        90206          Gi0/0/0/1
99.1.2.2
```

Example 9-56: LDP label binding for P-node prefix

```
RP/0/0/CPU0:xrvr-1#show mpls ldp bindings 1.1.1.4/32
1.1.1.4/32, rev 285
        Local binding: label: 90104
        Remote bindings: (3 peers)
            Peer                Label
            -----------------   ---------
            1.1.1.2:0           90204
            1.1.1.5:0           90504
```

The resulting MPLS forwarding entry for prefix 1.1.1.6/32 on Node1 is shown in Example 9-57. The local LDP label for prefix 1.1.1.6/32 on Node1 is 90106. The primary path to 1.1.1.6/32 is shown in lines 5 to 12 of the output and the TI-LFA repair path in lines 14 to 21. The repair path label stack (in line 18) contains three labels: {90504 30403 16006}. The top label 90504 is the outgoing LDP label to P-node Node4 as advertised by downstream LDP neighbor Node5; see the LDP label bindings for prefix 1.1.1.4/32 on Node1 shown in Example 9-56. The second and third labels, 30403 and 16006, are the result of the SR/LDP interworking replace operation in CEF. In this operation, CEF has replaced the unlabeled LDP label entries that are displayed in lines 12 and 13 of Example 9-55, with the last two labels in the IGP label stack shown on line 10 of Example 9-51: 30403 and 16006. The result of the replace operation is the repair label stack for LDP traffic destined for 1.1.1.6/32 as displayed on line 18 of Example 9-57.

Example 9-57: MPLS forwarding entry LDP label

```
 1| RP/0/0/CPU0:xrvr-1#show mpls forwarding prefix 1.1.1.6/32
detail
 2| Local  Outgoing    Prefix             Outgoing     Next
Hop          Bytes
 3| Label  Label       or ID
Interface                     Switched
 4| ------ ----------- ------------------ ------------ ---------
------ ------------
 5| 90106  90206       1.1.1.6/32         Gi0/0/0/1
99.1.2.2        0
 6|        Updated: Jul 11 15:53:18.961
 7|        Path Flags: 0x400 [  BKUP-IDX:0 (0xa0e8380c) ]
 8|        Version: 303, Priority: 15
 9|        Label Stack (Top -> Bottom): { 90206 }
10|        NHID: 0x0, Encap-ID: N/A, Path idx: 1, Backup path idx:
0, Weight: 0
11|        MAC/Encaps: 14/18, MTU: 1500
12|        Packets Switched: 0
13|
14|        90504       1.1.1.6/32         Gi0/0/0/0
99.1.5.5        0             (!)
15|        Updated: Jul 11 15:53:18.961
16|        Path Flags: 0x300 [  IDX:0 BKUP, NoFwd ]
17|        Version: 303, Priority: 15
18|        Label Stack (Top -> Bottom): { 90504 30403 16006 }
19|        NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx:
```

```
0, Weight: 0
          MAC/Encaps: 14/26, MTU: 1500
          Packets Switched: 0
          (!): FRR pure backup

```

Figure 9-22 illustrates the label stack of packets on the repair path. The complete path from Node11 to Node6, with the TI-LFA repair path activated, is represented unfolded in this illustration. An LDP labeled packet destined for Node6 arrives on PLR Node1 while protection is active. The label 90106 on this packet is the LDP label that Node1 allocated for 1.1.1.6/32. Node1 swaps the LDP incoming label 90106 with the outgoing Prefix-SID label 16006. On top of that Node1 imposes two additional labels: the Adjacency-SID label 30403 to steer the packet from Node4 to Node3, and the LDP label 90504 to bring the packet to Node4. LDP label 90504 is the LDP label advertised by Node5 for prefix 1.1.1.4/32.

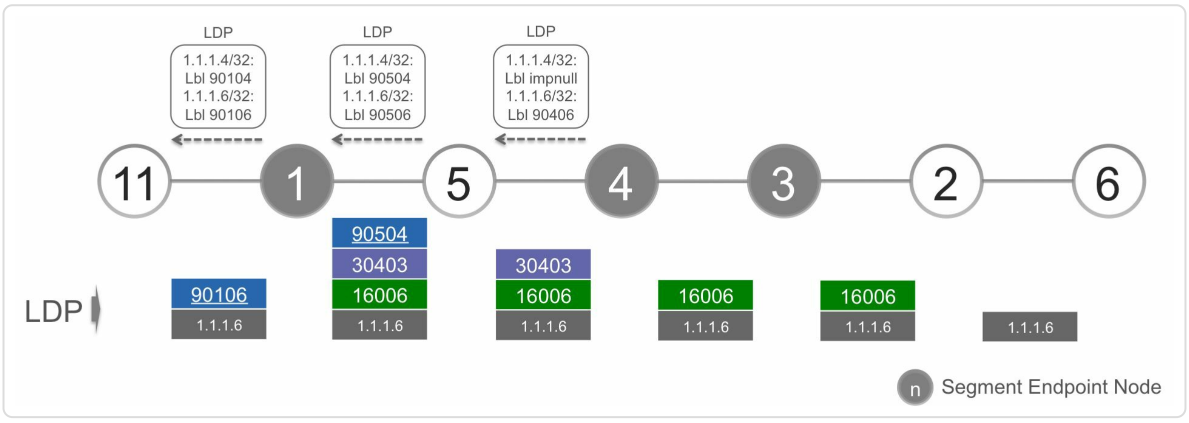

LDP
1.1.1.4/32: Lbl 90104 1.1.1.6/32: Lbl 90106
LDP
1.1.1.4/32: Lbl 90504 1.1.1.6/32: Lbl 90506
LDP
1.1.1.4/32: Lbl impnull 1.1.1.6/32: Lbl 90406
11
1
5
4
3
2
6
LDP
90106
1.1.1.6
90504
30403
16006
1.1.1.6
30403
16006
1.1.1.6
16006
1.1.1.6
16006
1.1.1.6
1.1.1.6
n Segment Endpoint Node

Figure 9-22: Label stack of packets on repair path

The repair path label stack that Node1 imposes on incoming unlabeled IP packets depends on Node1's label imposition preference. This label imposition preference indicates which label a node imposes on incoming unlabeled packets: the LDP outgoing label or the SR outgoing label. By default the imposition of an LDP outgoing label is preferred. This is the first case that is discussed here.

The CEF entry is the forwarding entry used for incoming unlabeled IP packets. The CEF entry for prefix 1.1.1.6/32 on Node1 is displayed in Example 9-58. The default label imposition preference is applied on Node1, also for the repair path. Therefore the CEF entry for 1.1.1.6/32 on Node1 shows the label stack for the repair path with an LDP outgoing label 90504 as top label (line 10). This is the same repair path label stack as for the LDP MPLS forwarding entry. Compare the label stack in line 10 of Example 9-58 with the one displayed on line 18 of Example 9-57. They are the same.

Example 9-58: Label imposition entry

```
 1 RP/0/0/CPU0:xrvr-1#show cef 1.1.1.6/32
 2 1.1.1.6/32, version 1626, internal 0x1000001 0x5 (ptr
0xa1434d74) [1], 0x0 (0xa14000e8), 0xa28 (0xa171c730)
 3  Updated May  4 09:08:31.589
 4  local adjacency 99.1.2.2
 5  Prefix Len 32, traffic index 0, precedence n/a, priority 15
 6    via 99.1.5.5/32, GigabitEthernet0/0/0/0, 13 dependencies,
weight 0, class 0, backup [flags 0x300]
 7     path-idx 0 NHID 0x0 [0xa110d2a4 0x0]
 8     next hop 99.1.5.5/32, P-node 1.1.1.4, Q-node 1.1.1.3
 9     local adjacency
10      local label 90106      labels imposed {90504 30403
16006}
11    via 99.1.2.2/32, GigabitEthernet0/0/0/1, 13 dependencies,
weight 0, class 0, protected [flags 0x400]
12     path-idx 1 bkup-idx 0 NHID 0x0 [0xa0e83c94 0x0]
13     next hop 99.1.2.2/32
14      local label 90106      labels imposed {90206}
```

The repair path can be verified with MPLS traceroute. The repair path's label stack can be specified on the command line when using the nil-fec keyword in the command. The MPLS traceroute output as collected on Node1 is displayed in Example 9-59. Note that the explicit-null label at the bottom of the label stack is present because of the nil-fec functionality.

Example 9-59: Traceroute of repair path label stack

```
RP/0/0/CPU0:xrvr-1#traceroute mpls nil-fec labels
90504,30403,16006 output interface Gi0/0/0/0 nexthop 99.1.5.5

Tracing MPLS Label Switched Path with Nil FEC with labels
[90504,30403,16006], timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output
interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC
mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx
label,
  'P' - no rx intf label prot, 'p' - premature termination of
LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

  0 99.1.5.1 MRU 1500 [Labels: 90504/30403/16006/explicit-null
Exp: 0/0/0/0]
L 1 99.1.5.5 MRU 1500 [Labels: implicit-
null/30403/16006/explicit-null Exp: 0/0/0/0] 0 ms
L 2 99.4.5.4 MRU 1500 [Labels: implicit-null/16006/explicit-
null Exp: 0/0/0] 10 ms
L 3 99.3.4.3 MRU 1500 [Labels: 16006/explicit-null Exp: 0/0] 0
ms
L 4 99.2.3.2 MRU 1500 [Labels: implicit-null/explicit-null Exp:
0/0] 20 ms
! 5 99.2.6.6 10 ms
```

With the default label imposition preference, the repair path label stack imposed on incoming unlabeled packets is the same as the one used for

LDP labeled traffic. When PLR Node1 is configured to prefer SR label imposition then the repair path label stack for unlabeled packets is the same as used for incoming SR labeled traffic. In Example 9-60, the operator configures the SR label imposition preference on Node1 and then verifies the CEF entry of prefix 1.1.1.6/32. Comparing the CEF entry with the output in Example 9-51, the local label has changed to the Prefix-SID label 16006 and the imposed labels have changed for both primary path and repair path. They both contain only SR labels.

Not that the label imposition preference configuration does not affect the repair path label stack for labeled packets. The MPLS forwarding entry for the LDP labeled packets destined for 1.1.1.6/32 stays unchanged compared to Example 9-57.

Example 9-60: Repair path label imposition entry with sr-prefer configured

```
RP/0/0/CPU0:xrvr-1#conf
RP/0/0/CPU0:xrvr-1(config)#router isis 1
RP/0/0/CPU0:xrvr-1(config-isis)# address-family ipv4 unicast
RP/0/0/CPU0:xrvr-1(config-isis-af)#  segment-routing mpls sr-
prefer
RP/0/0/CPU0:xrvr-1(config-isis-af)#commit
RP/0/0/CPU0:xrvr-1(config-isis-af)#end
RP/0/0/CPU0:xrvr-1#
RP/0/0/CPU0:xrvr-1#show cef 1.1.1.6/32
1.1.1.6/32, version 544, internal 0x1000001 0x83 (ptr
0xa14343f4) [1], 0x0 (0xa13ffb24), 0xa28 (0xa1784394)
 Updated May 15 06:42:59.644
 local adjacency 99.1.2.2
 Prefix Len 32, traffic index 0, precedence n/a, priority 1
   via 99.1.5.5/32, GigabitEthernet0/0/0/0, 17 dependencies,
weight 0, class 0, backup (remote) [flags 0x8300]
    path-idx 0 NHID 0x0 [0xa110d250 0x0]
    next hop 99.1.5.5/32, P-node 1.1.1.4, Q-node 1.1.1.3
    local adjacency
     local label 16006      labels imposed {16004 30403 16006}
   via 99.1.2.2/32, GigabitEthernet0/0/0/1, 17 dependencies,
weight 0, class 0, protected [flags 0x400]
```

```
    path-idx 1 bkup-idx 0 NHID 0x0 [0xa0e8380c 0x0]
    next hop 99.1.2.2/32
     local label 16006      labels imposed {16006}
```

9.8.3 TI-LFA and SR/LDP Interworking

The previous section discussed how IP and LDP traffic are protected by TI-LFA. The assumption was that all nodes on the TI-LFA repair path had SR enabled. This requirement can be relaxed somewhat. As long as the *key nodes* on the repair path have SR enabled then SR/LDP interworking functionality takes care of the end-to-end labeled path.

Figure 9-23 shows the same network topology as Figure 9-21, but now Node2, Node5, and Node6 do not have SR enabled. They are LDP-only nodes in this example. The other nodes in the network have both SR and LDP enabled. A Mapping Server advertises a Prefix-SID 16006 for prefix 1.1.1.6/32. Since the topology did not change, the PLR Node1 uses the same link protecting repair path for destination Node6, via P-node Node4 and Q-node Node3.

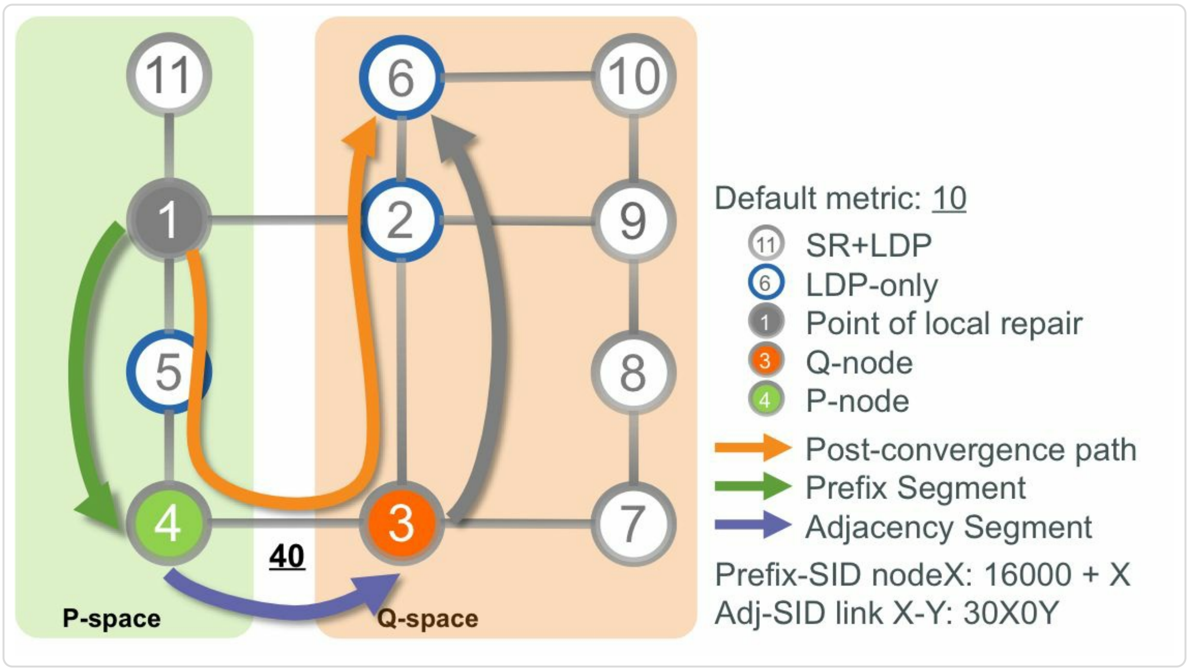


Figure 9-23: SR/LDP interworking on TI-LFA repair path

In this example the PLR Node1 must apply SR/LDP interworking functionality on the repair path since the downstream neighbor on the repair path, Node5, does not have SR enabled. Also, Node3 needs to apply SR/LDP interworking since its downstream neighbor Node2 towards Node6 has no SR enabled.

Example 9-61 shows the RIB entry for prefix 1.1.1.6/32 on PLR Node1. Notice that the RIB entry is marked as `labeled SR(SRMS)` (line 4), i.e. using a Prefix-SID advertised by a Mapping Server (SRMS). There is no outgoing SR label for the primary path (`Label: None` in line 18) since the downstream neighbor Node2 on the primary path has not enabled SR. Also the top label of the repair path is unlabeled. The labels of the repair path (line 10) are ordered top→bottom; the label value 0x100001 (1048577) on line 10 is an internal value representing *unlabeled*. The top label is unlabeled in this RIB entry because the downstream neighbor Node5 on the repair path has not enabled SR.

Example 9-61: SR/LDP interworking on repair path – RIB entry

```
 1 RP/0/0/CPU0:xrvr-1#show route 1.1.1.6/32 detail
 2
 3 Routing entry for 1.1.1.6/32
 4   Known via "isis 1", distance 115, metric 20, labeled
SR(SRMS), type level-2
 5   Installed Jul 12 13:52:17.309 for 00:06:13
 6   Routing Descriptor Blocks
 7     99.1.5.5, from 1.1.1.6, via GigabitEthernet0/0/0/0,
Backup (remote)
 8       Remote LFA is 1.1.1.4, 1.1.1.3
 9       Route metric is 0
10       Labels: 0x100001 0x76c3 0x3e86 (1048577 30403 16006)
11       Tunnel ID: None
12       Binding Label: None
```

```
        Extended communities count: 0
        Path id:65              Path ref count:1
        NHID:0x2(Ref:24)
      99.1.2.2, from 1.1.1.6, via GigabitEthernet0/0/0/1, Protected
        Route metric is 20
        Label: None
        Tunnel ID: None
        Binding Label: None
        Extended communities count: 0
        Path id:1       Path ref count:0
        NHID:0x3(Ref:20)
        Backup path id:65
    Route version is 0x31 (49)
    Local Label: 0x3e86 (16006)
    IP Precedence: Not Set
    QoS Group ID: Not Set
    Flow-tag: Not Set
    Fwd-class: Not Set
    Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD Type RIB_SVD_TYPE_LOCAL
    Download Priority 1, Download Version 770
    No advertising protos.
```

The LDP labels for destination 1.1.1.6/32 on Node1 are the same as displayed in Example 9-55. The output is repeated in Example 9-62.

Example 9-62: LDP forwarding entry

```
RP/0/0/CPU0:xrvr-1#show mpls ldp forwarding 1.1.1.6/32

Codes:
  - = GR label recovering, (!) = LFA FRR pure backup path
  {} = Label stack with multi-line output for a routing path
  G = GR, S = Stale, R = Remote LFA FRR backup

Prefix          Label   Label(s)       Outgoing     Next
Hop            Flags
                In      Out
Interface                        G S R
--------------- ------- -------------- ------------ -----------
-------- -----
```

```
1.1.1.6/32      90106   { 90504        Gi0/0/0/0
99.1.5.5        (!)     R
                          Unlabelled
(1.1.1.4)
                          Unlabelled }
(1.1.1.3)
                        90206          Gi0/0/0/1
99.1.2.2
```

The SR/LDP interworking functionality in CEF combines the SR labels and LDP labels to come to the SR MPLS forwarding entry for Prefix-SID label 16006 on Node1 as shown in Example 9-63. The primary path's outgoing label 90206 on line 9 is the LDP label advertised by Node2 for prefix 1.1.1.6/32. The top label 90504 for the repair path on line 18 is the LDP label advertised by Node5 for 1.1.1.4/32. These LDP labels in the Prefix-SID MPLS forwarding entry are the result of the SR/LDP interworking replace operation. CEF has replaced the unlabeled entries that were present in the RIB entry in Example 9-61, by their equivalent LDP labels.

Example 9-63: SR/LDP interworking on repair path – SR MPLS entry

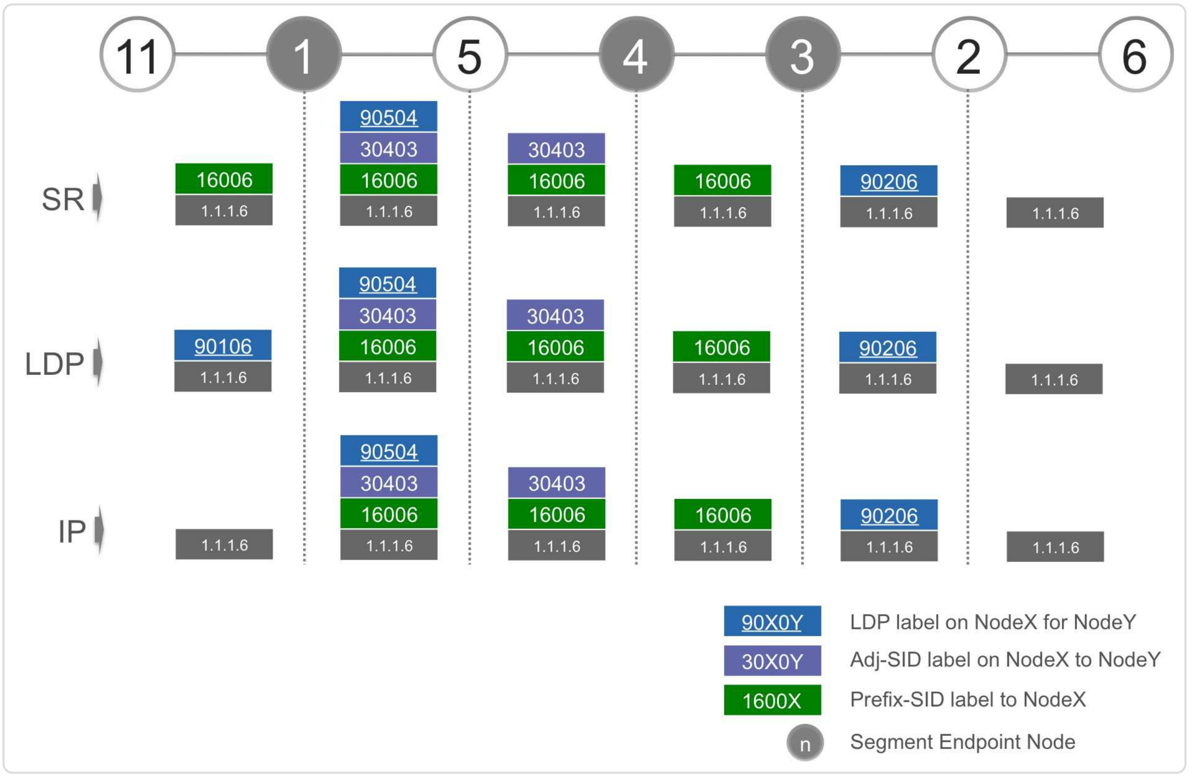
```
 1| RP/0/0/CPU0:xrvr-1#show mpls forwarding labels 16006 detail
 2| Local  Outgoing    Prefix             Outgoing     Next
Hop          Bytes
 3| Label  Label       or ID
Interface                     Switched
 4| ------ ----------- ------------------ ------------ ---------
------ ------------
 5| 16006  24021       SR Pfx (idx 6)     Gi0/0/0/1
99.1.2.2        0
 6|      Updated: Jul 12 13:52:17.259
 7|      Path Flags: 0x400 [  BKUP-IDX:0 (0xa0e8380c) ]
 8|      Version: 770, Priority: 15
 9|      Label Stack (Top -> Bottom): { 90206 }
10|      NHID: 0x0, Encap-ID: N/A, Path idx: 1, Backup path idx:
0, Weight: 0
11|      MAC/Encaps: 14/18, MTU: 1500
```

```
      Packets Switched: 0

        24008        SR Pfx (idx 6)     Gi0/0/0/0
99.1.5.5        0             (!)
      Updated: Jul 12 13:52:17.259
      Path Flags: 0x8300 [  IDX:0 BKUP, NoFwd ]
      Version: 770, Priority: 15
      Label Stack (Top -> Bottom): { 90504 30403 16006 }
      NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx:
0, Weight: 0
      MAC/Encaps: 14/26, MTU: 1500
      Packets Switched: 0
      (!): FRR pure backup

   Traffic-Matrix Packets/Bytes Switched: 0/0
```

Also Node3 applies the SR/LDP interworking functionality for Prefix-SID 16006, as illustrated in Figure 9-24. This illustration shows the unfolded representation of the path from Node11 to Node6 when the repair path is active. For incoming SR labeled traffic and LDP labeled traffic, the incoming top label is swapped with the Prefix-SID label 16006 of the destination. Then the same repair path label stack is imposed for both types of traffic: {90504, 30403}. Label 90504 is the LDP label advertised by Node5 for prefix 1.1.1.4/32. Label 30403 is the Adjacency-SID label of Node4 for the link to Node3. Node5 is the penultimate hop node for 1.1.1.4/32 and pops the LDP label. When the packet arrives on Node4 with top label 30403, Node4 pops the label and forwards the packet to Node3. Since Node2, the neighbor downstream from Node3, does not have SR enabled, Node3 swaps the Prefix-SID label 16006 with the LDP label 90206 that Node2 advertised for prefix 1.1.1.6/32. This is the SR/LDP interworking functionality on Node3.



11
1
5
4
3
2
6
SR
LDP
IP
90504
30403
16006
90106
90206
1.1.1.6
90X0Y
LDP label on NodeX for NodeY
30X0Y
Adj-SID label on NodeX to NodeY
1600X
Prefix-SID label to NodeX
n
Segment Endpoint Node

Figure 9-24: Label stack of packets on repair path

Unlabeled IP packets destined for 1.1.1.6 that arrive on Node1 will have the same repair path label stack imposed by Node1 as shown in Figure 9-24. Since 1.1.1.6/32 has an associated Prefix-SID 16006 advertised by a Mapping Server, Node1 can use label 16006 in the repair path's label stack to reach 1.1.1.6/32. The repair label stack does not depend on the label imposition preference configuration. The imposition preference would only influence the top label of the repair path's label stack. Since only the LDP outgoing label is available to reach P-node Node4, there is no choice but using that label in the repair label stack.

9.9 TI-LFA Load-Balancing on RepairPath

Although a single repair path is programmed for each individual destination prefix, TI-LFA uses the available ECMP in the network for the repaired traffic. The protected traffic uses all available ECMP for the Prefix Segments used on the repair path, thanks to the ECMP-awareness of Prefix-SIDs. For example from the PLR towards the P or PQ node and from the repair path's release point to the destination.

If multiple equivalent repair paths are possible, then TI-LFA statistically load-balances between them. This means that per destination, TI-LFA selects one of the available repair paths based on a hash function. If for example multiple equivalent P or PQ nodes are available, then TI-LFA statistically load-balances the protected destination prefixes over these different repair paths. Equivalent for the case of disjoint P and Q nodes; if there are multiple equal cost Q nodes from a P node, then again the PLR selects one (P, Q) pair for each destination prefix, based on a hash function.

Figure 9-25 illustrates the load balancing of traffic on the TI-LFA backup path. The topology is based on the topology in Figure 9-11, but two changes are applied. First, two nodes have been added: Node13 (between Node1 and Node4) and Node15 (between Node4 and Node2). Second, the link metrics for the links Node4-Node3 and Node4-Node13 are 30. This differs from the metric 40 used in Figure 9-11 for the link between Node4 and Node3. Using a metric 30 on these links does not change the P- and Q-spaces as computed by the PLR Node1 But with metric 30, the shortest path from Node4 to Node3 is via the direct link between them. Therefore Node4 can reach Node3 (and Node13) using its Prefix-SID. If the shortest path between P-node and Q-node is not via the direct link between them, then the Adjacency-SID must be used.

All nodes have enabled SR. The IGP link metrics in this topology are 10 except for the links between Node4 and Node3 and between Node4 and Node13 that have a metric 30. Node1 has TI-LFA link protection enabled and protects traffic to destinations Node6 and Node9, among others, against a failure of the link to Node2.

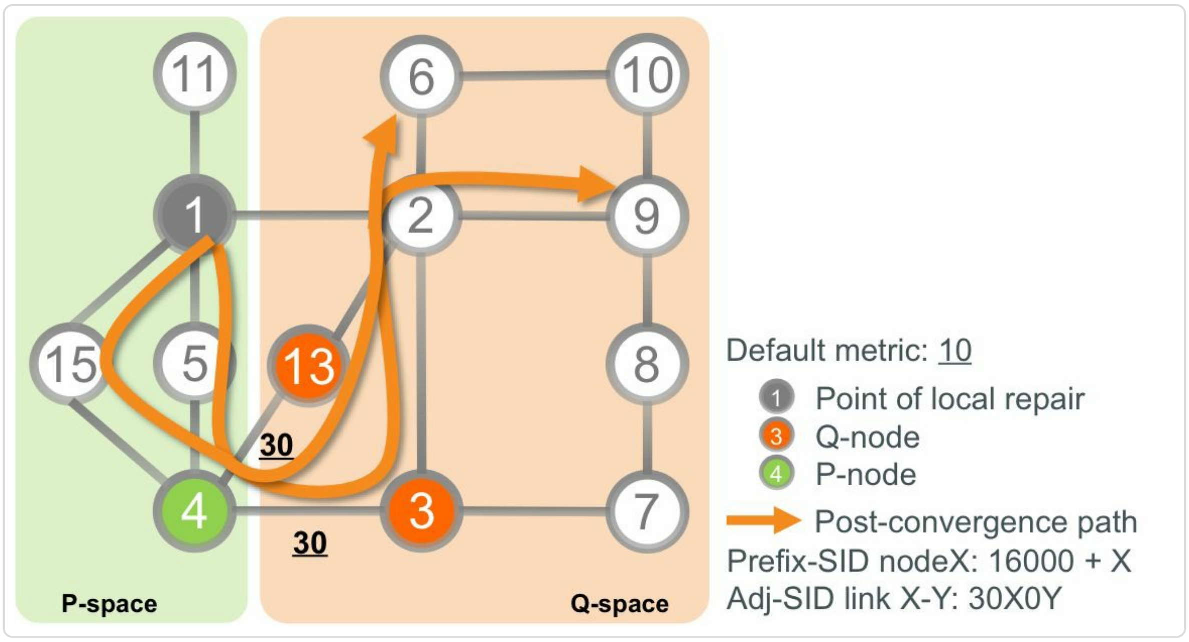


Figure 9-25: TI-LFA load-balancing on repair path

Node1 computes the TI-LFA repair path for destination Node6 and finds that two equally good repair paths are available: one via (P, Q)-pair (Node4, Node3) and another via (P, Q)-pair (Node4, Node13). The PLR installs a single repair path per destination. This repair path also includes the outgoing interface (OIF) and next-hop (NH). So, PLR Node1 has four repair paths to choose from:

- next-hop Node5, (P, Q)-pair (Node4, Node3)

- next-hop Node5, (P, Q)-pair (Node4, Node13)
- next-hop Node15, (P, Q)-pair (Node4, Node3)
- next-hop Node15, (P, Q)-pair (Node4, Node13)

The PLR Node2 tries to spread the protected traffic over all the available repair paths. It selects the repair path based on a hash computation (computed over destination prefix) to statistically load-balance the protected destinations over the available repair paths. As an example, Node2 picks the repair path via Node15 and (P, Q)-pair (Node4, Node13) to protect destination Node6 and to protect destination Node9 it selects the repair path via Node5 and (P, Q)-pair (Node4, Node3). See the ISIS output in Example 9-64. Notice that the entry for the Q-node Node13 is labeled `P node` (line 7). This indicates that the node before it (Node4) can reach Node13 via the post-convergence path using its Prefix-SID. This contrasts with the actual role of Node13: Q-node. The repair path for destination Node9 (1.1.1.9/32) goes via Node5, Node4, and Node3, as shown in Example 9-65.

Example 9-64: ISIS TI-LFA repair path for destination Node6

```
1 RP/0/0/CPU0:xrvr-1#show isis fast-reroute 1.1.1.6/32
2
3 L2 1.1.1.6/32 [20/115]
4      via 99.1.2.2, GigabitEthernet0/0/0/3, xrvr-2, SRGB Base:
16000, Weight: 0
5          Backup path: TI-LFA (link), via 99.1.15.15,
GigabitEthernet0/0/0/2 xrvr-15, SRGB Base: 16000, Weight: 0
6            P node: xrvr-4.00 [1.1.1.4], Label: 16004
7            P node: xrvr-13.00 [1.1.1.13], Label: 16013
8            Prefix label: 16006
```

Example 9-65: ISIS TI-LFA repair path for destination Node9

```
RP/0/0/CPU0:xrvr-1#show isis fast-reroute 1.1.1.9/32
```

```
L2 1.1.1.9/32 [20/115]
     via 99.1.2.2, GigabitEthernet0/0/0/3, xrvr-2, SRGB Base:
16000, Weight: 0
         Backup path: TI-LFA (link), via 99.1.5.5,
GigabitEthernet0/0/0/1 xrvr-5, SRGB Base: 16000, Weight: 0
           P node: xrvr-4.00 [1.1.1.4], Label: 16004
           P node: xrvr-3.00 [1.1.1.3], Label: 16003
           Prefix label: 16009
```

In the same topology, OSPF selects the repair path via Node5, Node4, and Node3 for destination Node6 (1.1.1.6/32). See the output in Example 9-66. The outgoing interface and next-hop are shown on line 13. The repair path for destination Node9 (1.1.1.9/32) goes via Node15, Node4, and Node13, as shown in Example 9-67.

Example 9-66: OSPF TI-LFA repair path for destination Node6

```
 1 RP/0/0/CPU0:xrvr-1#show ospf routes 1.1.1.6/32 backup-path
 2
 3 Topology Table for ospf 1 with ID 1.1.1.1
 4
 5 Codes: O - Intra area, O IA - Inter area
 6        O E1 - External type 1, O E2 - External type 2
 7        O N1 - NSSA external type 1, O N2 - NSSA external
type 2
 8
 9 O    1.1.1.6/32, metric 21
10        99.1.2.2, from 1.1.1.6, via GigabitEthernet0/0/0/3,
path-id 1
11            Backup path: TI-LFA, Repair-List:
12                                P node: 1.1.1.4
Label: 16004
13                                P node: 1.1.1.3
Label: 16003
14               99.1.5.5, from 1.1.1.6, via
GigabitEthernet0/0/0/1, protected bitmap 0000000000000001
15               Attributes: Metric: 71, SRLG Disjoint
```

Example 9-67: OSPF TI-LFA repair path for destination Node9

```
RP/0/0/CPU0:xrvr-1#show ospf routes 1.1.1.9/32 backup-path

Topology Table for ospf 1 with ID 1.1.1.1

Codes: O - Intra area, O IA - Inter area
       O E1 - External type 1, O E2 - External type 2
       O N1 - NSSA external type 1, O N2 - NSSA external type 2

O    1.1.1.9/32, metric 21
       99.1.2.2, from 1.1.1.9, via GigabitEthernet0/0/0/3,
path-id 1
           Backup path: TI-LFA, Repair-List:
                                P node: 1.1.1.4         Label:
16004
                                P node: 1.1.1.13        Label:
16013
              99.1.15.15, from 1.1.1.9, via
GigabitEthernet0/0/0/2, protected bitmap 0000000000000001
              Attributes: Metric: 71, SRLG Disjoint
```

In another example illustrated in Figure 9-26, Node20 is added to the topology between Node1 and nodes Node5 and Node15. In this example, the natural load-balancing nature of Prefix Segments is highlighted. PLR Node1 protects traffic to destinations Node6, Node9, and Node10, among others, against failure of the link to Node2. Node1 has two equally good repair paths to choose from:

- next-hop Node20, (P, Q)-pair (Node4, Node3)
- next-hop Node20, (P, Q)-pair (Node4, Node13)

PLR Node1 statistically load-balances the protected traffic over the two available repair paths, by selecting one of the repair paths per destination prefix. The traffic on the selected repair path is load-balanced over the two equal-cost paths from Node1 to P-node Node4 (via Node5 and via Node15) using the ECMP-awareness of the Prefix Segment to Node4. The traffic is also load-balanced over the equal-cost paths from the Q-node (Node3 or Node13) to the destination Node10 (via Node6 and via Node9) using the ECMP-awareness of the Prefix Segment to Node10. The load balancing over the equal-cost paths of a Prefix Segment is done using the regular data plane hashing functionality.

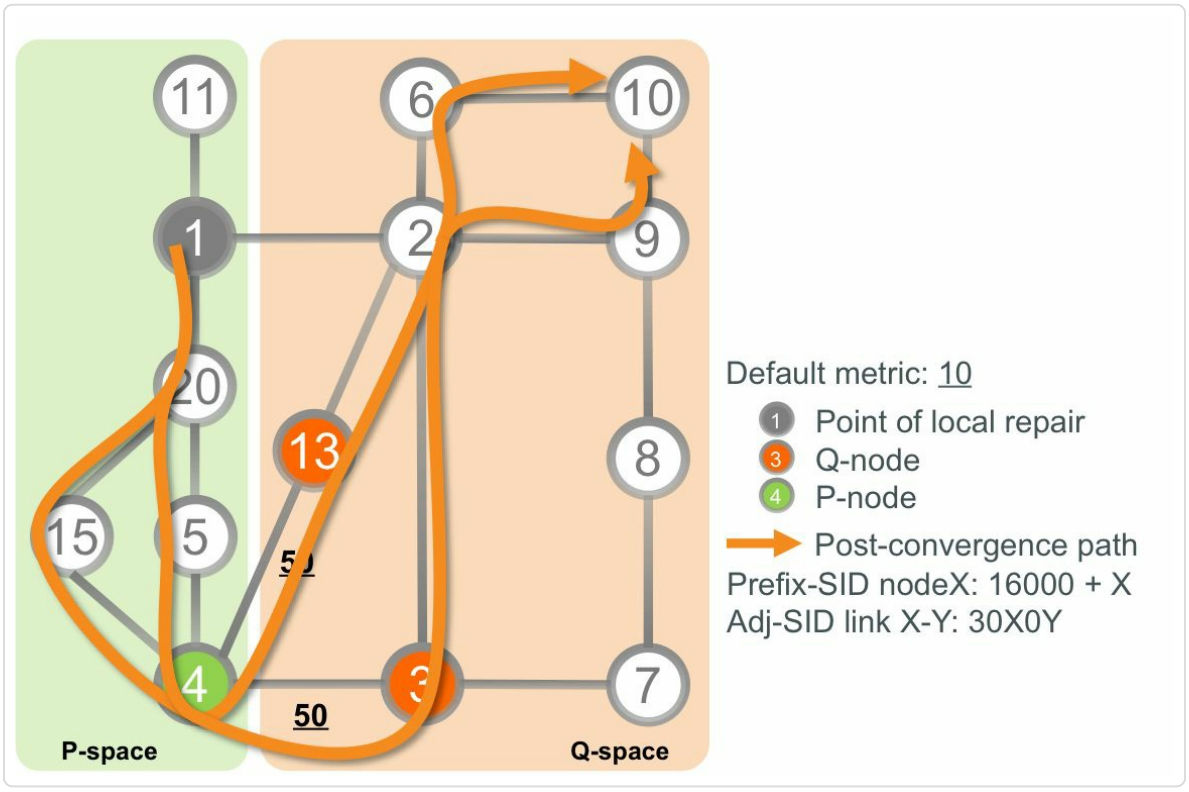


Figure 9-26: TI-LFA load-balancing on repair path

9.10 Microloop Avoidance

In this section, we will look at microloops that occur in IP/MPLS networks generally after convergence event. We will understand what are microloops and how they occur and then we will see how Segment Routing helps in implementing microloop avoidance.

9.10.1 An Overview of Microloops

When the topology of an IGP network changes (for example a link or node goes down) then the link-state IGP first distributes the information of the topology change to the other nodes in the network; the new link state advertisement is flooded in the network. Then each node in the network independently runs the same route computation algorithm and updates its forwarding tables with the new information. In this time period the forwarding tables on the nodes in the network are not consistent. Some nodes still forward traffic based on the old topology while other nodes have already updated their forwarding tables according to the new topology. Eventually, when all nodes have finished their forwarding table update, the network is said to be converged; the forwarding entries on all nodes are consistent and reflect the new topology.

The time period between the topology change and the update of a node’s forwarding table varies for each node for various reasons.

- The propagation of topology changes through the network introduces delays. The time it takes for a topology change notification to reach a node depends on the distance of the topology change to that node.
- The order in which each node updates prefixes in the forwarding table is not guaranteed to be the same.

- Control plane and data plane update speeds vary. It depends on CPU, ASIC, platform architecture, etc.

This inconsistent state of the forwarding tables of the nodes in the network following a topology change may result in traffic looping between nodes. The duration of the loops is limited by the convergence time of the slowest node.[5] These transient forwarding loops are called *microloops*, as they are typically sub-second in duration.

Microloops are a natural phenomenon in IP/MPLS networks that use hop-by-hop routing. They can occur for any topology change, good or bad, failure or restoration. They can occur close to the topology change or far from it.

The following example illustrates the occurrence of microloops. The network topology in Figure 9-27 has a default link metric 10. Traffic is flowing from Node6 to Node4, via the shortest path as indicated in the illustration. Another traffic stream from Node3 to destination Node4 is also indicated in the drawing. Now the link between Node1 and Node2 fails. In this first example, it is assumed that no fast-reroute protection is enabled on Node2. After the link failure, IGP on Node2 converges to the new topology and updates the forwarding entry to Node4 after 200 ms. At that time Node2 starts forwarding the traffic flow destined for Node4 via Node3. IGP on Node3 is somewhat slower and updates the forwarding entry to Node4 after 300 ms. Node3 now forwards traffic destined for Node4 directly to Node4. From the moment that Node2 uses the new forwarding path (after 200 ms) until the moment that Node3 installs the new forwarding entry (after 300 ms), traffic is looping between Node2 and Node3.

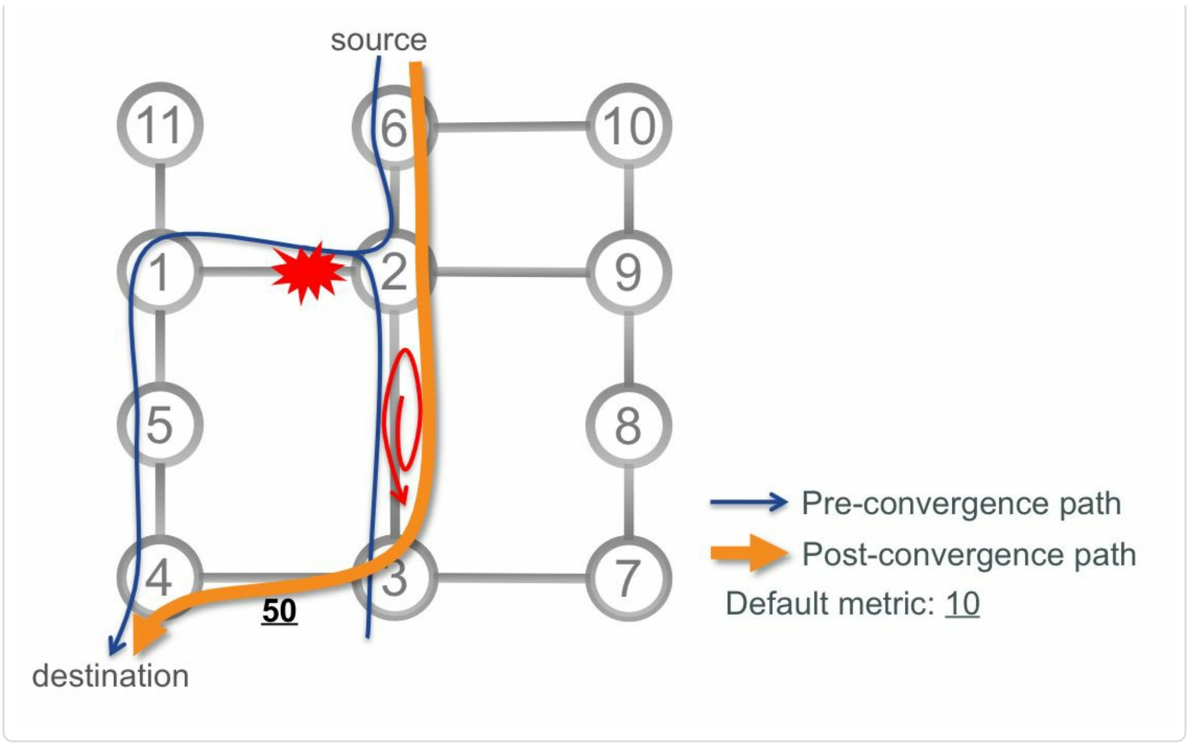


Figure 9-27: Microloop local to PLR

The timeline of the events as illustrated in Figure 9-28:

- 0 ms: the link between Node1 and Node2 goes down
- 200 ms: Node2 updates forwarding entry to Node6
- 300 ms: Node3 updates forwarding entry to Node6

During the time period 0 ms to 200 ms, Node2 drops the traffic. During the time period 200 ms to 300 ms (duration 100 ms) traffic loops between Node2 and Node3. At time 300 ms the traffic flow is restored. Assuming all packets in the microloop are lost or too late, then the loss of connectivity for the flow to Node6 following the link failure lasts from 0 ms to 300 ms. The microloop occurs during 1/3 of this time period (100 ms).

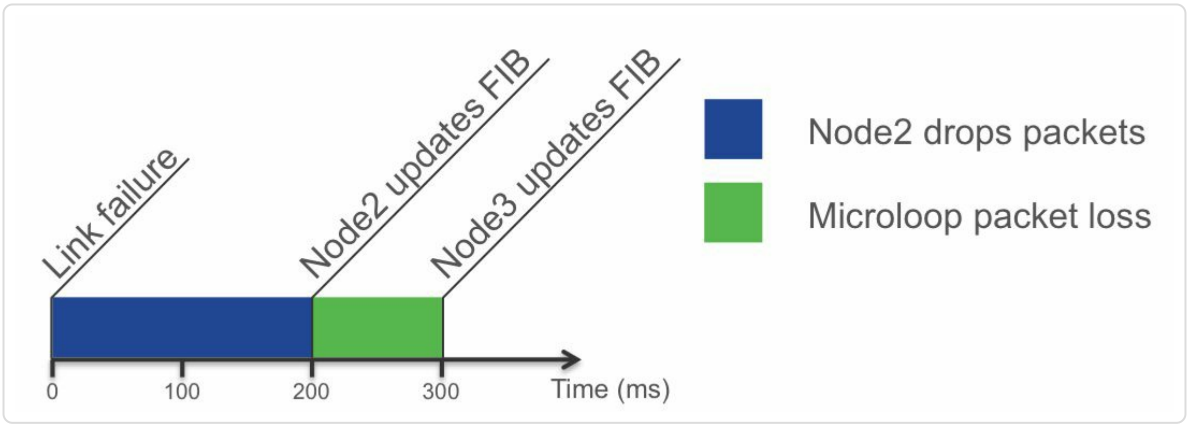


Figure 9-28: Microloop time line

Microloops have always existed in IP networks, but in the past they were not of major concern. They were too short comparing to overall network convergence time; the overall traffic loss during network convergence masked any traffic loss due to a routing loop. But after introducing FRR, traffic restoration after a failure typically happens in under 30 ms and the expectation is to see no more than 50 ms traffic loss when FRR is enabled. Traffic loss due to microloops now becomes noticeable. The problem is illustrated using IPFRR, but is independent of the FRR mechanism that is used. Notably it also applies to TE FRR protecting single-hop tunnels.

Microloops cause packet loss because the TTL of the packets in the loop eventually decreases to zero or because the link becomes saturated. Assume a microloop on a link with a one-way delay of 5 msec. If 20% of the link bandwidth was used before the event, then the link will be saturated due to the microloop after less than 20 ms, and packets will be dropped due to this saturation for the remaining duration of the microloop. The congestion occurs in both directions of the link. This link congestion caused by microloops can affect other traffic that would otherwise not be affected by the topology change; the congestion affects *all* traffic that traverses this link. The microloop illustrated in Figure 9-27 for example impacts traffic between Node6 and Node7; traffic that is otherwise not affected by the failure.

Going back to the example in Figure 9-27, now with TI-LFA link protection enabled on Node2. Node2 pre-installs the (loop-free) TI-LFA repair path for destination Node4 in the forwarding table. In this example, Node2 uses (P, Q)-pair (Node3, Node4) for the repair path to Node4. With this configuration, the sequence of events as illustrated in Figure 9-29, is as follows:

- 0 ms: the link between Node1 and Node2 goes down
- 25 ms: Node2 TI-LFA protection is activated
- 200 ms: Node2 updates forwarding entry to Node6 and removes the TI-LFA repair path
- 300 ms: Node3 updates forwarding entry to Node6

During the time period from 0 ms to 25 ms, Node2 drops the traffic. During the time period from 25 ms to 200 ms traffic reaches its destination via the TI-LFA repair path; during this time period there is no loss of connectivity. During the time period from 200 ms to 300 ms (duration 100 ms), traffic loops between Node2 and Node3. At time 300 ms the traffic flow is restored.

There are two distinct periods of packets loss in this example: one right after the failure (0 ms – 25 ms) and one after Node2 converges (200 ms – 300 ms). The total loss of connectivity for the flow to Node6 following the failure lasts 125 ms (25 ms + 100 ms). The microloop occurs during 4/5 of this time period.

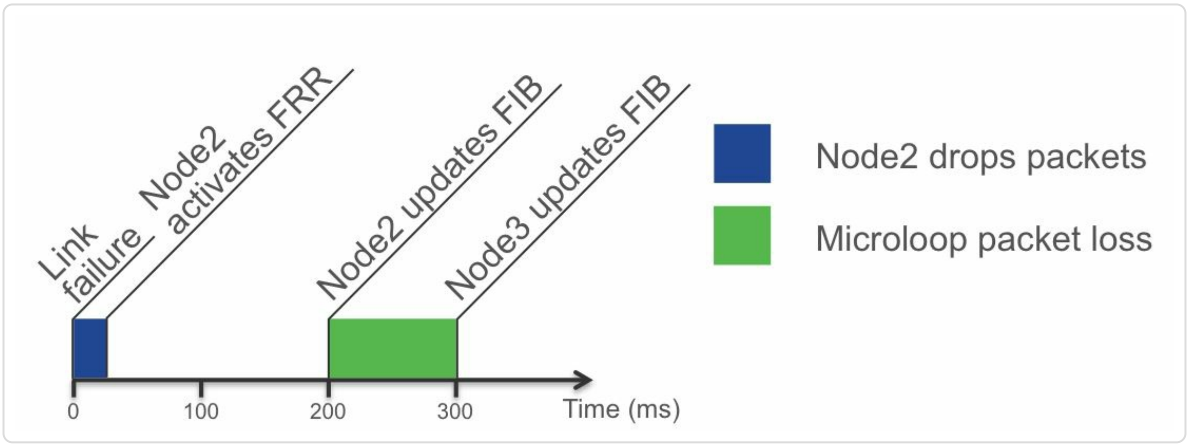


Figure 9-29: Microloop time line with FRR

9.10.2 Existing Microloops Avoidance Mechanisms

Since the introduction of LFA, a simple microloop avoidance mechanism exists in Cisco IOS XR. Using this mechanism the protecting node (PLR) delays updating its forwarding table to allow all other nodes in the network to converge to the new topology. The PLR computes the new paths, but delays installing its forwarding table. After this delay, the PLR updates its forwarding table and since the rest of the network is already in a consistent state, the microloop is prevented. The local microloop avoidance functionality is described in IETF draft-ietf-rtgwg-uloop-delay. If this local microloop avoidance functionality is enabled on Node2 in Figure 9-27,, then the sequence of events is as illustrated in Figure 9-30:

- 0 ms: the link between Node1 and Node2 goes down
- 25 ms: Node2 TI-LFA protection is activated
- 300 ms: Node3 updates forwarding entry to Node6
- 5025 ms: Node2 updates forwarding entry to Node6 and removes the TI-LFA repair path

During the time period 0 ms to 25 ms, Node2 drops the traffic. After this, Node2 steers the traffic on the (loop-free) repair path until Node2 updates its forwarding table after 5 seconds. Since Node3 has already updated its forwarding table, no loop occurs.

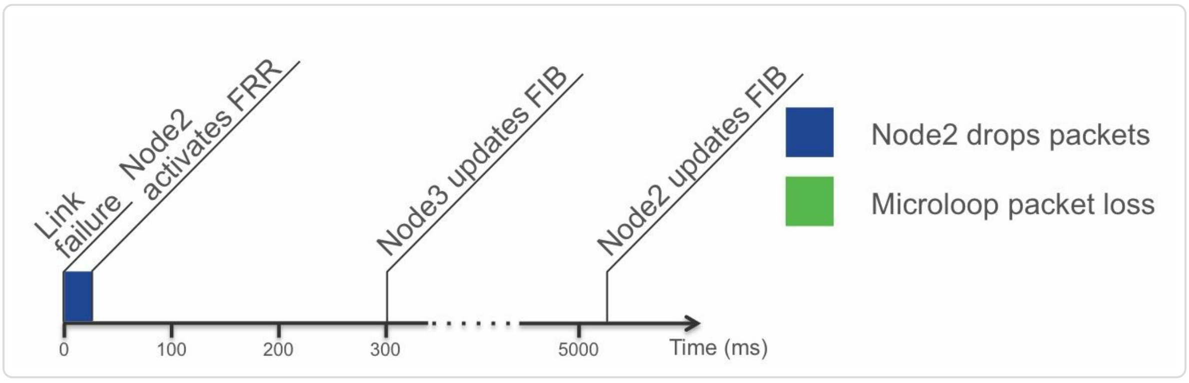


Figure 9-30: Microloop time line with FRR and local microloop avoidance

However, this local microloop avoidance scheme has limitations: it does not work for all microloops, and it only works for link down events. As an example, this mechanism does not work for a different failure in the topology; see Figure 9-31. Before the failure of the link between Node5 and Node4, the traffic between source Node5 and destination Node6 is as indicated by the line that is labeled *Pre-convergence path* in the illustration. Also the shortest path from Node3 to the destination is indicated. Then the link Node5-Node3 fails and the line that is labeled *Post-convergence path* indicates the path after the convergence.

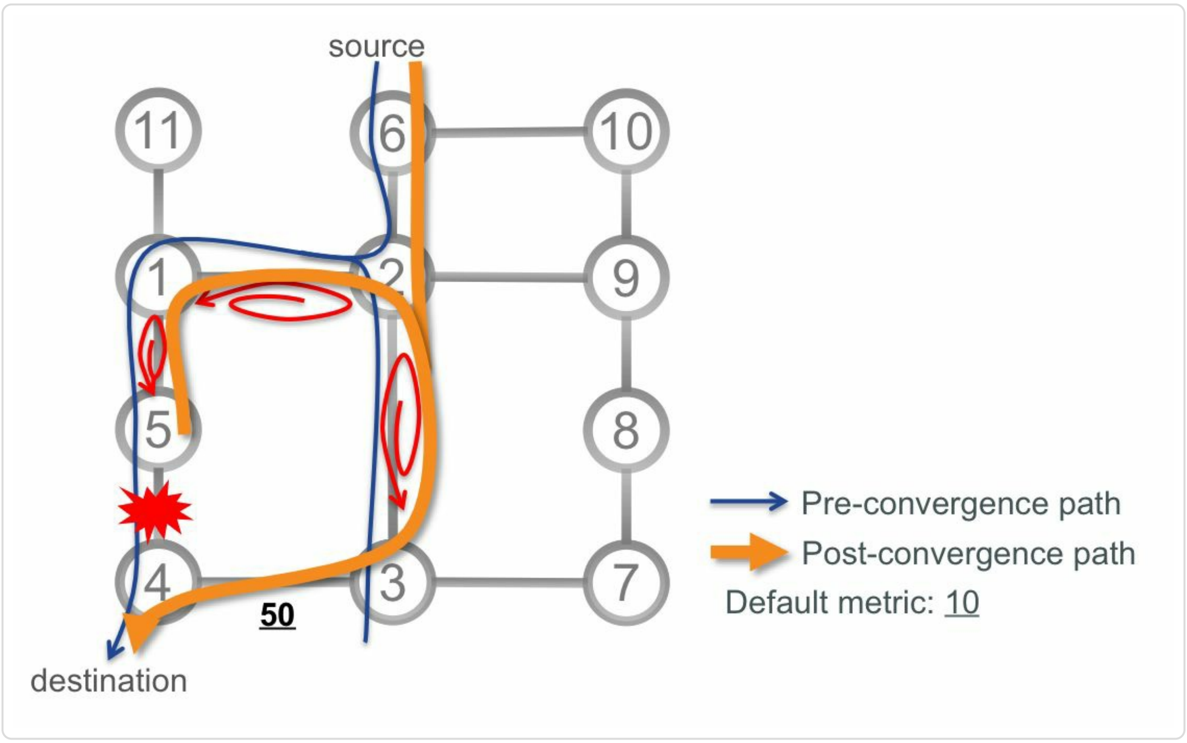


Figure 9-31: Microloops remote from PLR

Depending on the convergence order of the nodes, three microloops are possible, as indicated in the illustration. In the worst case scenario the sequence of events is as follows (see Figure 9-32):

- 0 ms: the link between Node4 and Node5 goes down
- 25 ms: Node5 TI-LFA protection is activated
- 200 ms: Node1 updates its forwarding table
- 250 ms: Node2 updates its forwarding table
- 300 ms: Node3 updates its forwarding table
- 5025 ms: Node5 updates its forwarding table (local microloop avoidance)

Traffic loss occurs during the first 25 ms, until Node5 activates TI-LFA protection. Since Node5 steers the traffic on the loop-free repair path, no traffic loss occurs until Node1 updates its forwarding table. At 200 ms a microloop starts between Node1 and Node2. That microloop is resolved when Node2 updates its forwarding table at time 250 ms. But at the same time a new microloop starts between Node2 and Node3. This microloop lasts until Node3 updates its forwarding table at time 300 ms. Sometime later, at around 5 s, Node5 updates its forwarding table. Local microloop avoidance can only prevent one of the possible microloops from occurring: the microloop between Node1 and Node5. In this example, the traffic stream experiences loss of connectivity during 125 ms, of which 100 ms due to microloops.

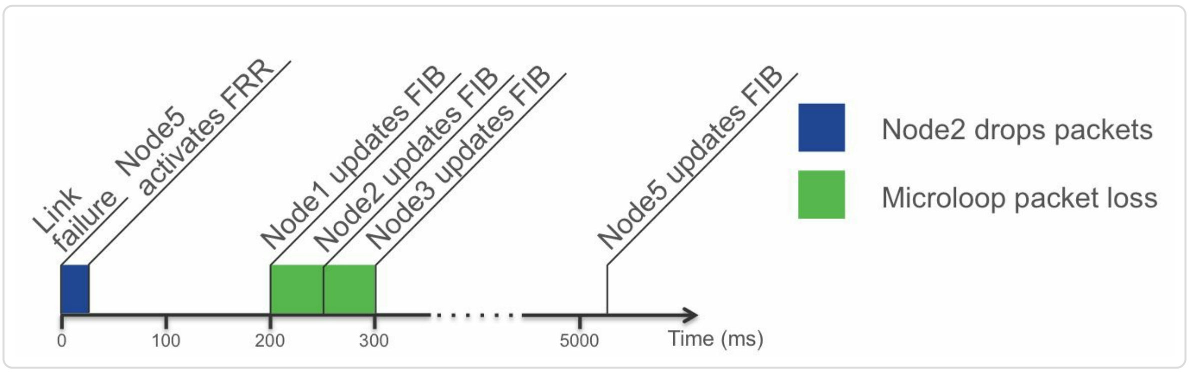


Figure 9-32: Remote microloop time line with FRR

So far, this section only covered microloops occurring after a link failure. As mentioned before, microloops can also occur after bringing up a link. In an example using the same topology, the link between Node4 and Node5 is restored; see Figure 9-33. The potential microloops impacting traffic going from Node6 to Node4 are indicated in the drawing. A possible sequence of events is as follows (see Figure 9-34:

- 0 ms: The link between Node4 and Node5 is restored
- 200 ms: Node2 updates its forwarding table
- 250 ms: Node1 updates its forwarding table

- 300 ms: Node5 updates its forwarding table

In this sequence of events, a microloop occurs between Node2 and Node1 in the time period 200 ms to 250 ms. Node2 steers packets to the destination towards Node1, but Node still sends the packets following the old topology. After Node1 updates its forwarding table, a microloop occurs between Node1 and Node5 during the period 250 ms to 300 ms. After Node5 updates its forwarding table, the microloop is resolved. In this example, the traffic experienced 100 ms packet loss due to microloops; while for link restoration no loss is desired and also expected.

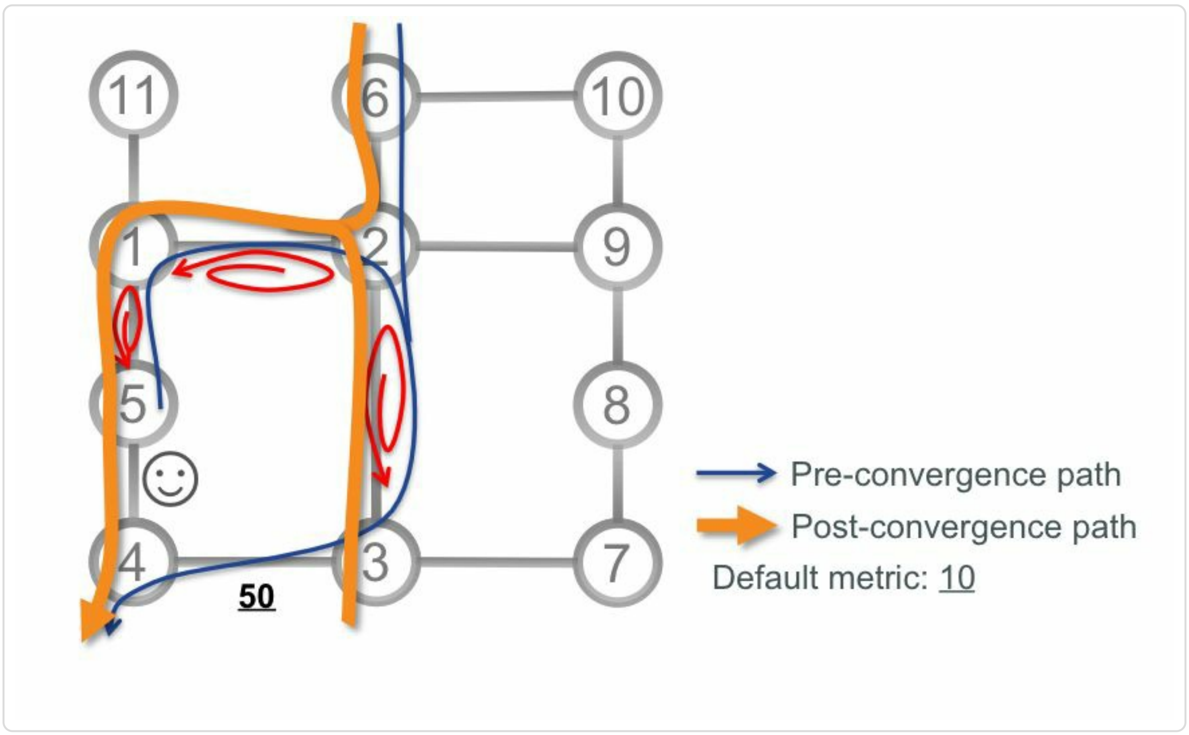


Figure 9-33: Microloops for link restoration

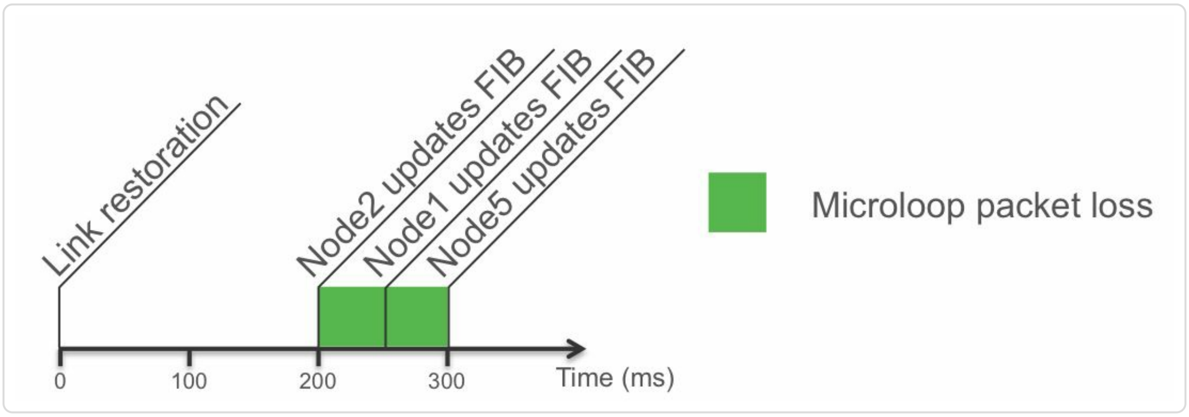


Figure 9-34: Link restoration microloop time line

Microloops have been discussed and investigated for several years. Multiple solutions have been proposed to avoid microloops, but none provided a complete solution that was simple to implement and operate. IETF RFC 5715 describes a number of proposed solutions for the problem. The local microloop avoidance solution described in IETF draft-ietf-rtgwg-uloop-delay provides a partial solution for link down cases.

9.10.3 Microloop Avoidance with Segment Routing

The explicit source routed traffic steering of Segment Routing provides a simple and complete solution for the microloop problem. When a topology change occurs in the network and a converging node computes that microloops are possible on the post-convergence path, then that node uses SR to prevent these microloops from occurring. The node temporarily steers the traffic on the post-convergence path using an explicit segment list to a loop free release point (much like in case if TI-LFA). When there is no longer any risk for microloops then the node switches to the regular forwarding without the explicit Segment list. Packets then follow their regular shortest path in the new topology without any risk for microloops.

The microloop avoidance convergence process consists of two phases. Upon a topology change, node N analyses the topology change and finds that microloops are possible on the post-convergence path to destination D. Therefore node N applies the following two-stage convergence process for destination D:

Stage 1: Node N computes a loop-free SR path that is tailored along the post-convergence path. For a period in time, node N steers the traffic to D via using that explicit loop-free post-convergence path. The period of time is at least the worst-case convergence time of a node, network-wide.

Stage 2: After the Stage 1 timer elapses, N installs the regular post-convergence forwarding path for destination D. So, node N no longer imposes any additional segments to ensure loop-freeness of the path.

As an example, the SR microloop avoidance functionality is enabled in the same topology as was used earlier in this section; see Figure 9-35. Before the link failure, the traffic from Node6 to Node4 follows the path that is labeled *Pre-convergence path* in the illustration. When the link between Node4 and Node5 fails, Node5 (and Node4) activates TI-LFA protection and floods the topology change notification (IGP link-state advertisement) in the network. Thanks to TI-LFA FRR protection, traffic loss following the failure is limited to less than 50 ms.

All nodes compute the new Shortest Path Tree (SPT). Node6 computes the post-convergence path to Node4 (labeled *Post-convergence path* in the illustration) and finds that a microloop is possible on that post-convergence path between Node2 and Node3, as indicated in the illustration. Therefore, Node6 computes a Segment List to steer the traffic to Node4 in a loop-free manner. The algorithm to determine the possibility of microloops and for computing a loop-free explicit path to avoid microloops is proprietary and a local function on the Node 6 – hence not publicized. Node6 temporarily imposes the Segment List {Prefix-SID(Node3), Adj-SID(L3-4)} on the packets destined for Node4. This is the first stage of the convergence process. From then the packets already traverse the post-convergence path, which is the most desired path since it is the path that the traffic will follow after the network has fully converged. The Prefix-SID(Node3) in the Segment List brings the packets to Node3 in a loop-free manner since the path from Node6 to Node3 is not affected by the topology change. The Adjacency-SID in the Segment List then steers the traffic from Node3 to Node4 and from there the traffic goes to its destination unaffected by a microloop.

The other nodes in the network also apply the microloop avoidance process. For example, Node2 imposes the Segment List {Adj-SID(L3-4)} and sends the packets on the outgoing interface to Node3. Node11 imposes the Segment List {Prefix-SID(Node3), Adj-SID(L3-4)}. The path from Node11 to Node3 is loop-free since it is not affected by the topology change.

In the second stage of the convergence process, the nodes switch to their regular forwarding paths. They no longer impose any additional segments on the packets since at that time the network is free from potential microloops. This second stage does not need to happen synchronously on all nodes. For example, when Node2 switches to the regular forwarding then Node6 can still use its explicit post-convergence path for some time. Or vice versa. The paths do not affect each other.

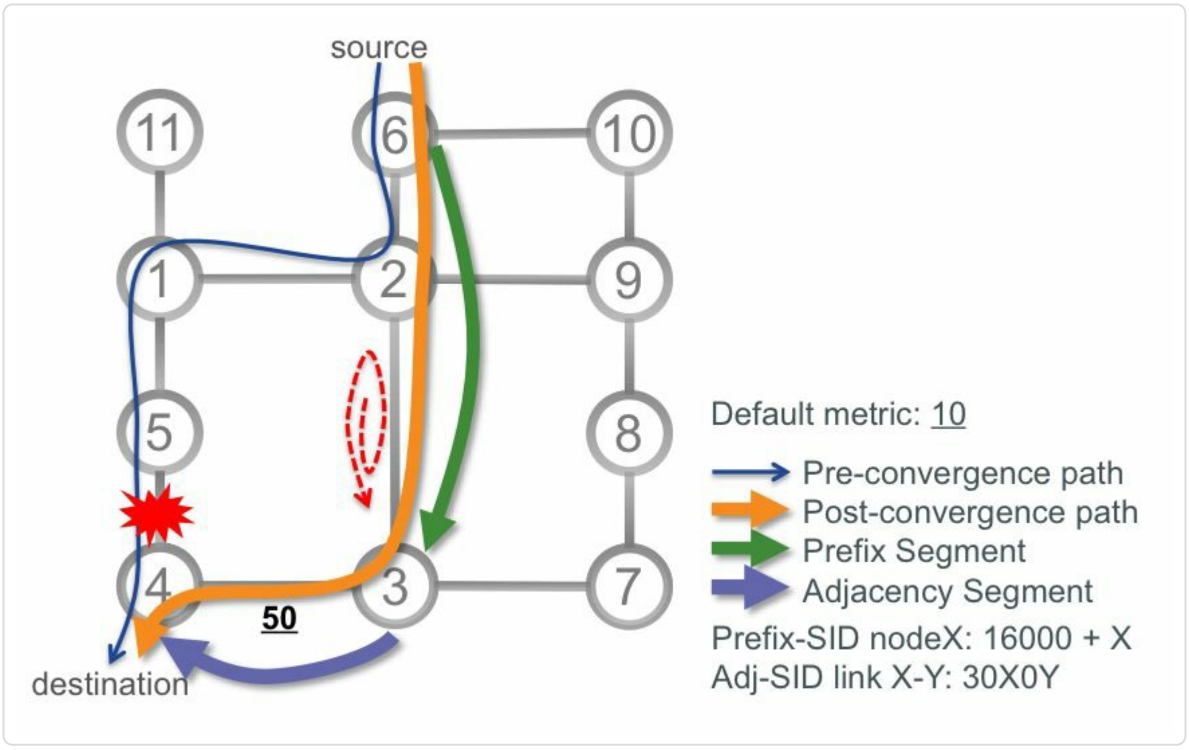


Figure 9-35: SR microloop avoidance

The Segment Routing microloop avoidance functionality prevents microloops from occurring for single link events. These events include link failure and restoration as well as link metric changes.

SR microloop avoidance is a local behavior. Each node individually computes and applies the required loop-free SR paths. No signaling is required for these source routed SR paths on the other nodes in the network. Basic SR forwarding functionality (Prefix-SID, Adj-SID) is required for the nodes on the post-convergence path.

Microloop avoidance is applied for SR and LDP labeled traffic as well as for unlabeled IP traffic.

It is not required to do a full network upgrade to benefit from this functionality. SR microloop avoidance is incrementally deployable, with incremental benefits. Traffic passing through nodes that have this functionality enabled benefits from it. Nodes that do not have this functionality enabled will still cause the microloops as they caused before the functionality was introduced, unless the local behavior of another node that supports the microloop avoidance functionality prevents these loops from occurring.

The IOS XR implementation of the SR microloop avoidance functionality instantiates SRTE Policies for the temporarily explicit post-convergence paths. This functionality is equivalent to the TI-LFA functionality described in section 9.4.3.

At the time of writing this book, the microloop avoidance functionality was not yet released for Cisco IOS XR. We will provide an update and more details on this topic in a future revision of this book.

"Deployment of LFA could be considered as a quick win in an IP network that does not mandate a guaranteed traffic protection: it is very simple to deploy (usually one CLI command), is very simple to understand, and has an insignificant scaling impact while providing an almost good coverage in usual topologies. Implementing remote LFA adds an additional level; while it is also simple to deploy (one CLI command), it may be harder to operate, as the RLFA candidate would need to accept TLDP sessions from a PLR (additional feature required everywhere in the network). With RLFA it is hard to predict what would be the best RLFA candidate for a particular destination from a particular PLR without using a simulation tool. Depending on the design and network size, a particular RLFA candidate may be used by plenty of PLRs, thus leading to massive TLDP sessions setup which may add scaling considerations.

Customers' expectations in term of quality of experience are growing with the critical nature of their applications. As MPLS transport is the foundation of most carriers' services, it must be efficient and robust, also because bad things can happen in a network and the network must dynamically adapt without disrupting the customers' applications. In this area, micro-loops have always been a pain for IP/MPLS networks by breaking fast-reroute or creating micro-congestions. I was interested in micro-loop prevention for many years by investigating, evaluating, and implementing multiple solutions.

But all those past solutions were only partial or too complex to be deployed in a live network: the work on the *local delay* solution was a first quick win that helped for sure but that could not be considered as a definitive solution as it does not solve the remote microloops. Solutions like *Ordered FIB* introduced too much complexity: trying to order node computation in a network where by design nodes are independent of each other, was not the right way to go.

Now, thanks to the Segment Routing building blocks, we have the technology to easily build loop-free paths in the network in a simple way. I had the chance to play with and evaluate in depth an early code running SR microloop avoidance, and I can confirm that it works and provides again a very simple solution to prevent microloops for good and bad events: a single CLI command on your routers and you will prevent microloops."

— Stéphane Litkowski

9.11 Summary

- TI-LFA provides sub-50msec link, node and SRLG protection with 100% coverage.
- TI-LFA is simple to operate and understand. The functionality is contained within the IGP and no additional protocols or signaling is required.
- The repair path is automatically computed by the IGP, no specific tuning is required.
- By using the post-convergence path as backup path, it prevents transient congestion and suboptimal routing on the backup path.
- TI-LFA can be incrementally deployed; it is a local functionality and it also protects LDP and IP traffic.
- SR Microloop avoidance prevents microloops to occur following a topology change (link up, link down, link metric change). Traffic is temporary steered on the post-convergence path that is made loop-free by using SR's traffic steering capabilities.

9.12 References

- [draft-francois-rtgwg-segment-routing-uloop] Francois, P., Filsfils, C., Bashandy, A., and Litkowski, S., "Loop avoidance using Segment Routing",draft-francois-rtgwg-segment-routing-uloop (work in progress), June 2016, https://datatracker.ietf.org/doc/draft-francois-rtgwg-segment-routing-uloop.
- [draft-francois-spring-segment-routing-ti-lfa] Filsfils, C., Bashandy, A., Decraene, B., and Francois, P., "Topology Independent Fast Reroute using Segment Routing", draft-francois-spring-segment-routing-ti-lfa, (work in progress), April 2015, https://datatracker.ietf.org/doc/draft-francois-spring-segment-routing-ti-lfa.
- [FRANCOIS] Pierre François, "Improving the Convergence of IP Routing Protocols", PhD thesis, Université Catholique de Louvain, October 2007, http://inl.info.ucl.ac.be/system/files/pierre-francois-phd-thesis_0.pdf.
- [ietf-rtgwg-uloop-delay] Litkowski, S., Decraene, B., Filsfils, C., and Francois, P., "Microloop prevention by introducing a local convergence delay", draft-ietf-rtgwg-uloop-delay (work in progress), June 2016, https://datatracker.ietf.org/doc/draft-ietf-rtgwg-uloop-delay.
- [MPLSWC14] Bruno Decraene, Stéphane Litkowski, Orange, "Topology independent LFA – Orange use-case and applicability", MPLS SDN World Congress, Paris, March 2014, http://www.slideshare.net/StephaneLitkowski/mpls-sdn-2014-topology-independant-lfa.
- [MPLSWC15] Stéphane Litkowski, Orange, "SPRING interoperability

testing report", MPLS SDN World Congress, Paris, March 2015, http://www.slideshare.net/StephaneLitkowski/mpls-sdn-2015-spring-interoperability-testing.

- [MPLSWC16] Stéphane Litkowski, Orange, "Avoiding microloops using segment routing", MPLS SDN World Congress, Paris, March 2016, http://www.slideshare.net/StephaneLitkowski/mpls-sdn-2016-microloop-avoidance-with-segment-routing-63809004.
- [RFC4202] Kompella, K., Ed., and Y. Rekhter, Ed., "Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4202, DOI 10.17487/RFC4202, October 2005, https://datatracker.ietf.org/doc/rfc4202.
- [RFC4203] Kompella, K., Ed., and Y. Rekhter, Ed., "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4203, DOI 10.17487/RFC4203, October 2005, https://datatracker.ietf.org/doc/rfc4203.
- [RFC5286] Atlas, A., Ed., and A. Zinin, Ed., "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, DOI 10.17487/RFC5286, September 2008, https://datatracker.ietf.org/doc/rfc5286.
- [RFC5307] Kompella, K., Ed., and Y. Rekhter, Ed., "IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 5307, DOI 10.17487/RFC5307, October 2008, https://datatracker.ietf.org/doc/rfc5307.
- [RFC5715] Shand, M. and S. Bryant, "A Framework for Loop-Free Convergence", RFC 5715, DOI 10.17487/RFC5715, January 2010, https://datatracker.ietf.org/doc/rfc5715.
- [RFC6571] Filsfils, C., Ed., Francois, P., Ed., Shand, M., Decraene, B.,

Uttaro, J., Leymann, N., and M. Horneffer, “Loop-Free Alternate (LFA) Applicability in Service Provider (SP) Networks”, RFC 6571, DOI 10.17487/RFC6571, June 2012, https://datatracker.ietf.org/doc/rfc6571.

- [RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, “Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)”, RFC 7490, DOI 10.17487/RFC7490, April 2015, https://datatracker.ietf.org/doc/rfc7490.
- [RFC7916] Litkowski, S., Ed., Decraene, B., Filsfils, C., Raza, K., Horneffer, M., and P. Sarkar, “Operational Management of Loop-Free Alternates”, RFC 7916, DOI 10.17487/RFC7916, July 2016, https://datatracker.ietf.org/doc/rfc7916.

---

[1] The order in which prefixes are updated is *random*. But prefixes can be classified in priority classes, with prefixes in a higher priority class being updated before those in a lower priority class. By default, host prefixes have a higher priority than non-host prefixes.

[2] An SR Traffic-Engineering (SRTE) Policy (sometimes called SRTE Encapsulation Policy) is the forwarding construct that imposes one or more segments on packets that are steered into it. Although it has different properties, it is often seen as the SR counterpart of an RSVP-TE tunnel.

[3] Package Installation Envelopes (PIE) is a software package that is installed on top of the base Cisco IOS XR software package and is used to enable certain functionality that is not included in the base software package.

[4] The TI-LFA IETF draft-francois-rtgwg-segment-routing-ti-lfa proposes another method for the case a prefix-SID follows the Adj-SID in the label stack: pop Adj-SID and forward to the Prefix-SID that was underneath. This is not implemented in Cisco IOS-XR implementation at the time of writing this book..

[5] This is worst case. The duration of a microloop is in fact the delta time between the updates of the forwarding entry of the nodes participating in the microloop.

10 LARGE SCALE INTERCONNECT WITH SEGMENT ROUTING

The MPLS label space "only" contains about 1 million labels (2^{20}). Unlike IP, there is no concept of label summarization and default label. Each node needs a its own specific label for the forwarding. In large-scale networks with millions of endpoints this would mean millions of entries in the forwarding tables. This is undesirable for the low cost Data Center switches or Service Provider Metro Ethernet access nodes. Segment Routing MPLS can scale the network to support hundreds of thousands of nodes and tens of millions of physical underlay endpoints, while only requiring a few tens of thousands entries in the forwarding tables.

HIGHLIGHT: Bigger SRGB is possible and often the better option

In this section, we use the default SRGB as an "extreme" illustration. We show that even with a very small SRGB of 8000 global labels, we can scale to a network of hundreds of thousands of nodes and tens of millions of physical underlay endpoints.

Obviously, if a specific deployment is made of 20k nodes, one can simply use an SRGB of 20k global labels (likely 32k for future growth) and use a classic design where each node receives its own globally-unique prefix SID.

The SRGB can conceptually be allocated much bigger size (e.g. 256k entries).

The hierarchical hyper-scale design presented in this section is to be leverage either when the number of nodes exceed the MPLS label space size or when the FIB capabilities of the network nodes need to be minimized.

Aspects of this new design model are also the subject of the IETF draft-filsfils-spring-large-scale-interconnect

With transitions like Internet of Things (IoT) and increase in mobile devices, the number of devices getting connected to the network is exploding. Networks also need to scale as a result to handle this growth. This design model enabled by Segment Routing allows the network to scale while still keeping the operations simple.

10.1 Applicability Considerations

In most networks, there is benefit in the simplicity that classic designs provide and use a homogenous SRGB across the domain. Traditional IGP design concepts of areas/levels help with the scalability and stability aspects. Note that SRGB can grow to accommodate to ensure that every router in these networks can get their own Node-SIDs and also for different service endpoints.

However, there are certain large scale networks that are split into multiple domains for scale due to the sheer number of routers and service endpoints that need to be handled. The Seamless MPLS Architecture (refer draft-ietf-mpls-seamless-mpls) is one such reference design that is used for large aggregation networks. Another aspect to consider is the scalability of the routing platforms involved, especially the access/aggregation nodes in an aggregation network or the top-of-the-rack and leaf switches in the data center. These platforms being lower cost devices as compared to the core and edge routing platforms, have lower FIB scale. There is therefore a need to ensure end to end IP/MPLS connectivity from any to any service endpoint but at the same time not blow up the number of forwarding entries required especially in the leaf or access nodes. The design model described in this chapter can be most suitably applied to the interconnection of massive-scale Data Centers or large Service Provider aggregation networks.

10.2 Reference Design

The reference design network topology is illustrated in Figure 10-1. The network is structured in leaf domains (Leaf1, Leaf2, …) interconnected by a central Core domain. Only two leaf domains are shown in the illustration. Each domain runs Segment Routing with its own independent routing protocol (e.g.: IS-IS, OSPF, BGP). Each leaf domain Leaf*k* connects to the Core domain with two or more nodes called X*k*a and X*k*b. Each node X runs two independent SR routing protocols: one in the leaf domain and one in the core domain.

A common SRGB of [16000-23999] is assumed across all of the domains. Any other common SRGB choice is possible. We further assume that the sub-range [16000-17999] of the SRGB is solely used to provide prefix segments in the core domain while the sub-range [18000-23999] is reused to provide prefix segments in *any* leaf domain. Any other choice to divide the SRGB in sub-ranges is possible. Note that these sub-ranges only exist administratively; all the nodes still allocate the full SRGB.

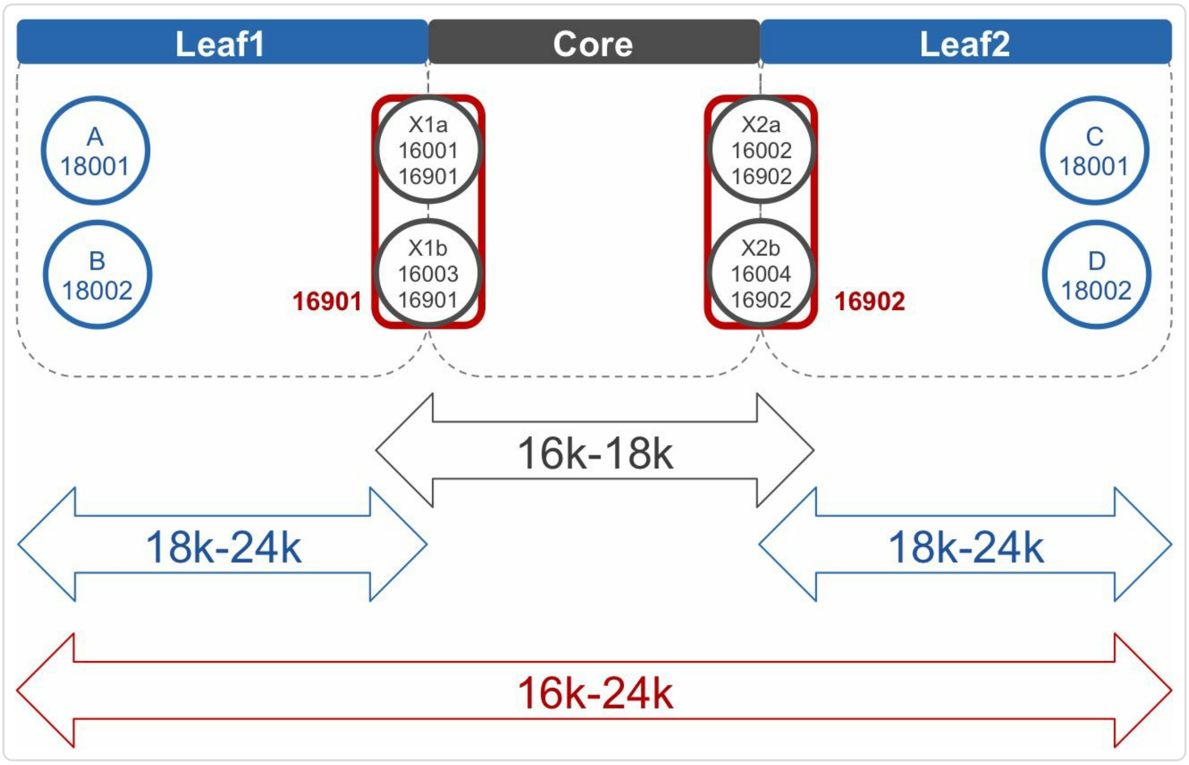


Figure 10-1: Large-scale interconnect reference design

The same SRGB sub-range [18000-23999] is reused to provide prefix segments in *any* leaf domain. For example, the operator allocated the same Prefix-SID 18001 to nodes A and C of the leaf domains Leaf1 and Leaf2 respectively. The SRGB sub-range [16000-17999] is not reused, but dedicated to allocate Prefix-SIDs for Core domain nodes. For example, the operator allocated Prefix-SID 16001 to node X1a in the Core domain. This Prefix-SID is unique across all the domains.

The operator allocates two Prefix-SIDs from the Core domain's SRGB sub-range to each node X: one that uniquely identifies the node (Node-SID) while the other is shared by them (i.e Anycast-SID) and identifies the pair of X nodes interconnecting the leaf domain Leaf*k* to the core domain. Node X1a in Figure 10-1 has Prefix-SID 16001 and Anycast-SID 16901, while node X1b has Prefix-SID 16003 and Anycast-SID 16901. The Anycast-SID provides load-balancing and simple node redundancy. By sending traffic to an Anycast-SID, the traffic will be handled by the closest node that advertises this Anycast-SID, or load-balanced over multiple nodes if they are at equal distance from the source. If one of the X nodes fails, then the other seamlessly takes over thanks to the property of the Anycast-SID. The Anycast-SIDs are also important for the Segment Routing Traffic Engineering Policies that are deployed in this model, but this is something that we will cover in the next part of the book.

All the prefixes (and their Prefix-SIDs) of the X nodes are redistributed from the core domain into the leaf domains, and no other prefix is redistributed from the core domain into the leaf domains. No prefix is redistributed from a leaf domain to the core domain. Therefore the forwarding table of an internal node within the core domain does not hold any entry for segments in the range [18000-23999] (the leaf domains' SRGB sub-range). This is illustrated in Figure 10-2.

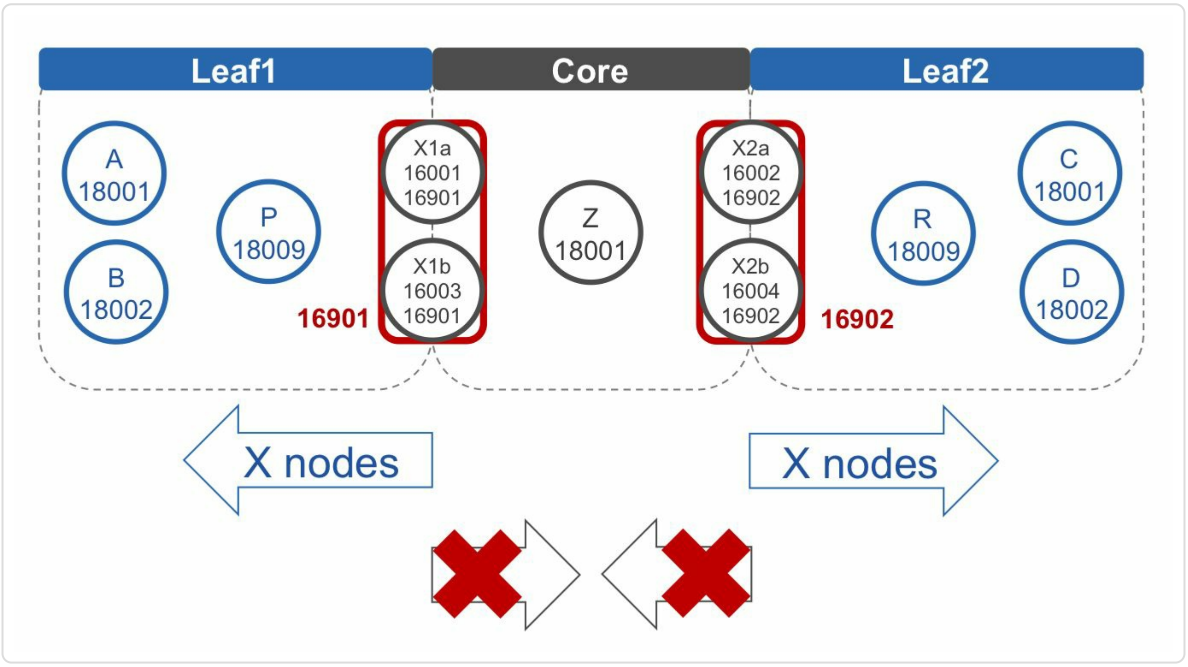


Figure 10-2: Redistribution between domains

A node in a leaf domain only has forwarding entries for all the segments in the local leaf domain and for the prefix segments towards all the X nodes in the network. For example, node A of leaf domain L1 has a forwarding entry for Anycast-SID 16902, which leads to the pair X2a and X2b and for example also for Prefix-SID 16005, which leads to node X2a.

10.2.1 Interconnecting Nodes

With this design, any leaf node can interconnect with any other leaf node. And any endpoint can connect to any endpoint. To start, let us assume that a leaf node can get the SID for the remote leaf itself and the intermediate hops on the path to reach the remote leaf with their SIDs via some "out-of-band" mechanism.

Figure 10-3 illustrates interconnecting nodes.

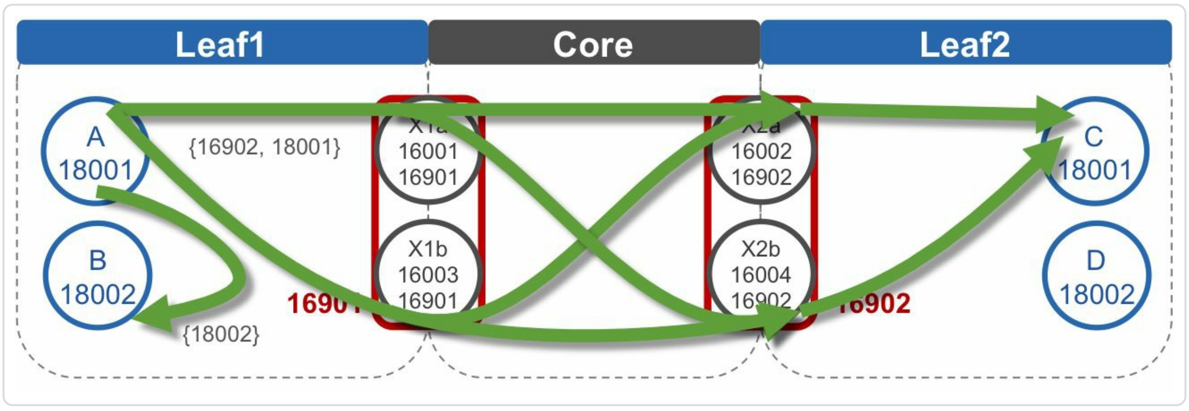


Figure 10-3: Interconnecting nodes

Node A uses the Segment List {18002} to reach node B via the shortest path. SID 18002 is the Prefix-SID of node B. This is an example of intra-domain connectivity; both nodes are located in the same leaf domain.

To provide inter-domain connectivity, the Segment List contains a *domain ZIP code*, which identifies a domain, and a *street name*. The *domain ZIP* *code* is the Anycast Segment of the border nodes to the destination domain. The *street name* is the Prefix Segment of the destination node in that domain. Node A can reach node C on the shortest path via *any* intermediate node X using the Segment List {16902, 18001}. SID 16902 is the Anycast-SID of the X nodes that interconnect domain Leaf 2 to the Core domain (nodes X2a and X2b), and SID 18001 is the Prefix-SID of node C in Leaf2 domain.

For node A to reach node C on the shortest path via a specific node, for example via node X2a, node A can use the Segment List {16002, 18001}. SID 16002 is the Prefix-SID of node X2a. This is an example of inter-domain connectivity; both nodes are located in a different leaf domain.

This book does not intend to cover how a source node derives and programs the Segment List to reach a remote node or endpoint. Static configuration or a centralized controller are obvious candidates. This is handled using Segment Routing TE concepts which is a subject to be covered in the next book.

10.2.2 Interconnecting Endpoints

Figure 10-6 shows endpoints attached to node A and node C. Such endpoint can for example be a Network Interface Device (NID) or a Virtual Machine (VM). The leaf node allocates an Adjacency-SID for each attached endpoint. The Adjacency Segments are locally significant to the node; therefore each leaf node can allocate the same Adjacency-SIDs. Continuing the comparison with the postal service, the Adjacency Segment is the *house number*. An endpoint can be attached to more than one leaf node (multi-homed).

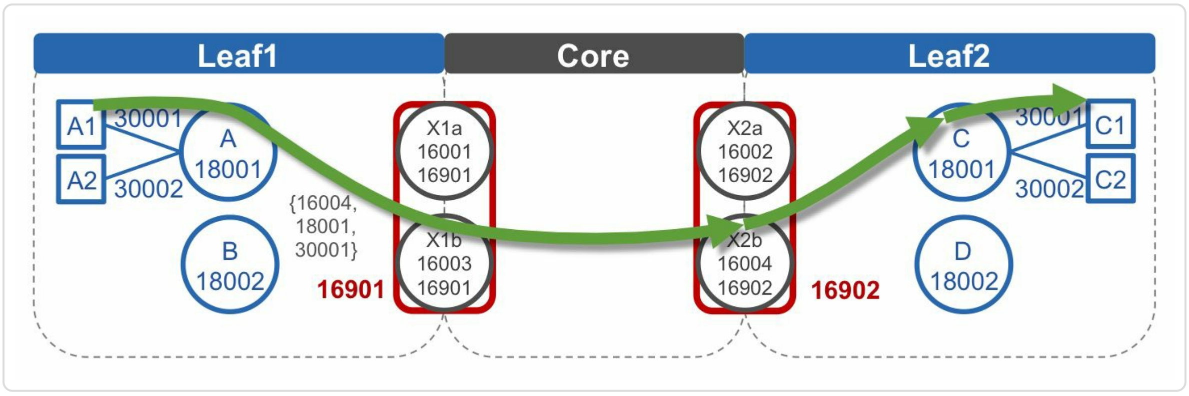


Figure 10-4: Interconnecting endpoints

To reach an endpoint attached to a leaf node, the local segment of the leaf node for that endpoint must be added to the Segment List. For example, to reach endpoint C1, connected to node C, from endpoint A1, connected to node A, the Segment List {16004, 18001, 30001} can be used. The first two segments steer the packets to leaf node C via node X2b, then node C's local SID 30001 steers the packets to endpoint C1. Similar to the previous example, if there is no requirement to pass through X2b specifically, then the Anycast-SID of the pair (X2a, X2b) could have been used as the first Segment of the path.

The discovery of the service endpoints via some "out-of-band" mechanism is again outside the scope of this book and will be covered in a next book.

10.3 Design Options

10.3.1 Leaf and Core Domains Sizing

The operator may choose to not redistribute *any* routes between the domains, not even the routes for the X nodes. This design option decreases the amount of forwarding entries that are required on the nodes in the leaf domains. In that case one more segment must be added to the Segment List that expresses the end-to-end path. For example, the path from node A to node C through any intermediate X node uses the Segment List {16901, 16902, 18001}. SID 16901 is the Anycast-SID of the X nodes that interconnect domain Leaf1 to the Core domain (nodes X1a and X1b). This SID brings the packets to node X1a or X1b. SID 16902 is the Anycast-SID of nodes X2a and X2b, and brings the packets to one of these nodes. SID 18001 is the Prefix-SID of destination node C.

We still assume that an "out-of-band" mechanism exists that provides the leaf with the connectivity information, similar to the previous cases.

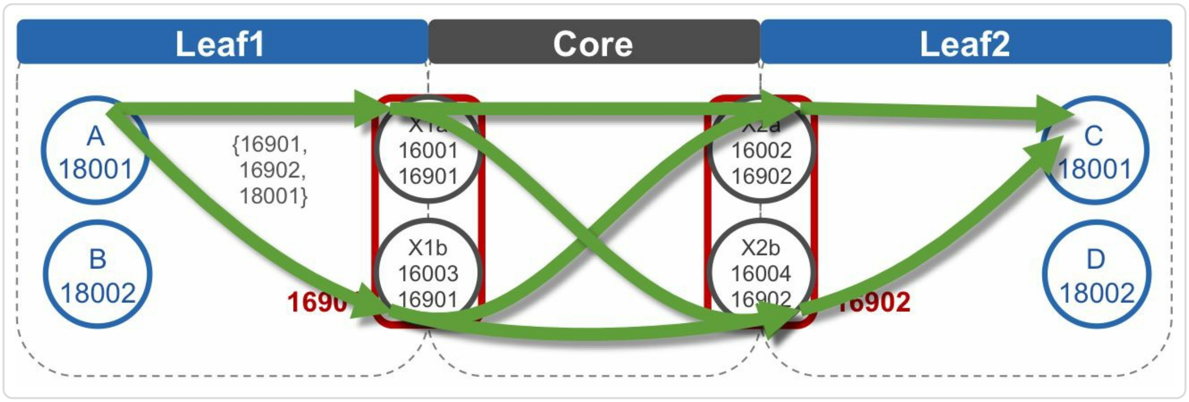


Figure 10-5: Connectivity without redistribution

10.3.2 Sub-Leaf Domains

To further increase scale, a third level of hierarchy called "Sub-Leaf" can be introduced in the reference design. These sub-leaf domains are connected to the leaf domains.

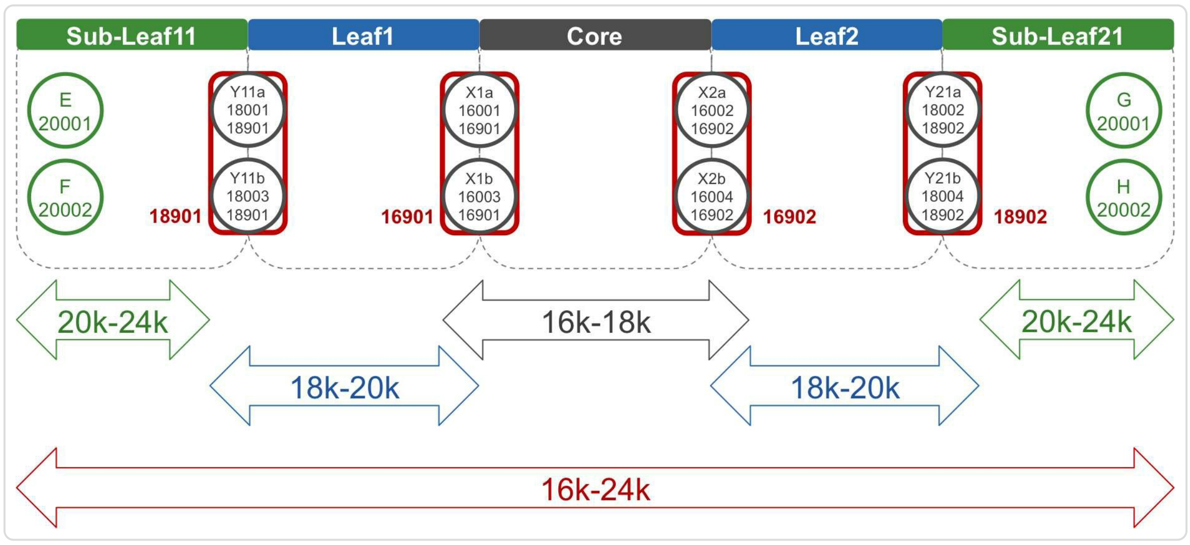


Figure 10-6: Sub-leaf domains

In Figure 10-6, sub-leaf domains SubLeaf11 and SubLeaf21 have been added to the network topology. The sub-leaf domains are connected to the leaf domains via two (or more) Y nodes. The SRGB sub-space [18000-23999] that was initially allocated for the leaf domains is split into two sub-spaces: [18000-19999] for the allocation of SIDs in the leaf domains and [20000-23999] for the allocation of SIDs in the sub-leaf domain. An Anycast-SID and a Prefix-SID from the leaf SRGB sub-range are allocated for each node Y. For example, Prefix-SID 18001 and Anycast-SID 18901 are allocated for Y21a. Each node within a sub-leaf domain receives a unique Prefix-SID from that level's SRGB sub-range (e.g. G receives 20001).

Node E can reach node G on the shortest path via *any* intermediate node X and *any* intermediate node Y by using the Segment List {16902, 18901, 20001}. SID 16902 is the Anycast-SID of nodes X2a and X2b). SID

18901 is the Anycast-SID of nodes Y21a and Y21b, and SID 20001 is the Prefix-SID of node E.

10.3.3 Traffic Engineering

So far, only shortest path transport was discussed. However, SR provides the ability to steer packets on any end-to-end path through the network. This makes it possible to steer service traffic on a path that satisfies the requirements of the service, without creating any additional state on the nodes other than the source node.

SR can be used to traffic-engineer flows within each leaf or core domain. See the topology in Figure 10-7. For example, a flow from node A in domain Leaf1 to node X1a within the same Leaf1 domain could be steered via node P using the SRTE Policy {18009, 16001}. Similarly, a flow from node X1a to node X2a within the core domain could be steered via node Z with the SRTE Policy {16009, 16002}.

Similarly, a flow can be engineered across domains. For example, a flow from node A in domain Leaf1 to node D in domain Leaf2 could be steered via path A → B → X1a → X2b → R → D using the SRTE Policy {18002, 16001, 16004, 18009, 18002}.

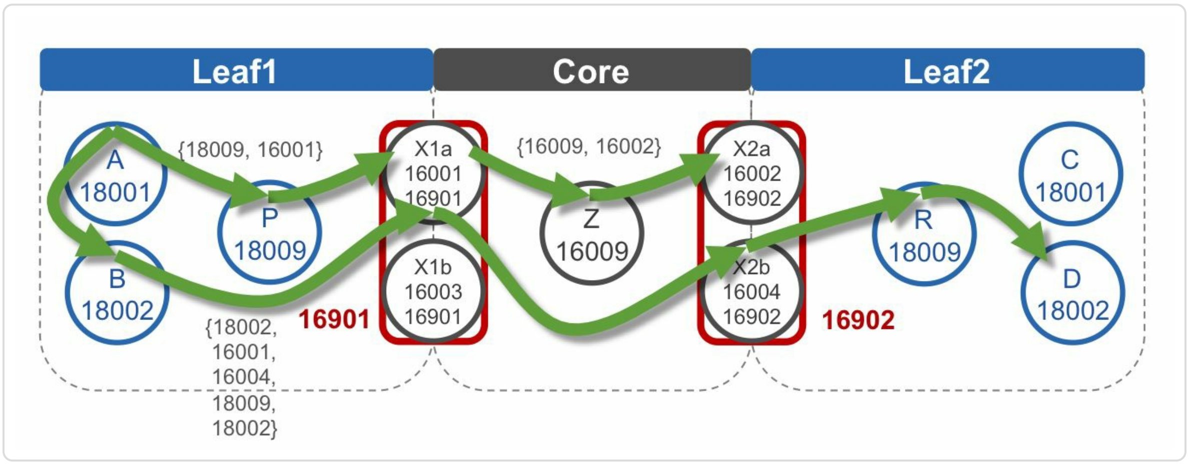


Figure 10-7: Traffic Engineering illustration

SR Traffic Engineering is not in scope of this first book and will be covered in the next part.

HIGHLIGHT

End-to-End MPLS connectivity in large scale networks The large scale interconnect design with SR provides end-to-end MPLS connectivity in a simple and scalable manner. The real benefit is the ability to do inter-domain Traffic Engineering with SR (to be covered in the next part of this book) also in a similar simple and scalable manner with controllers.

10.4 Scale Example

This section analyses the scaling properties of an example network following the reference design described above. The network consists of a single core domain and 100 leaf domains. The default SRGB range [16000-23999] is carved into two sub-ranges: [16000-17999] for the core domain and [18000-23999] for the leaf domains.

The scaling numbers are illustrated in Figure 10-8. Each leaf domain contains 6000 nodes. Assuming that a Prefix-SID is allocated for each node, then each leaf domain contains 6000 Prefix-SIDs. This is the maximum for the SRGB sub-range that was reserved for the leaf domains. If larger leaf domains are desired then the SRGB can be enlarged to accommodate that.

Each node in a leaf domain has 500 endpoint attached to it, thus 500 Adj-SIDs are programmed on each leaf node; one per attached endpoint. In total there are 3 million (= 6000 * 500) endpoints per leaf domain. With 100 leaf domains this makes a total of 300 million endpoints.

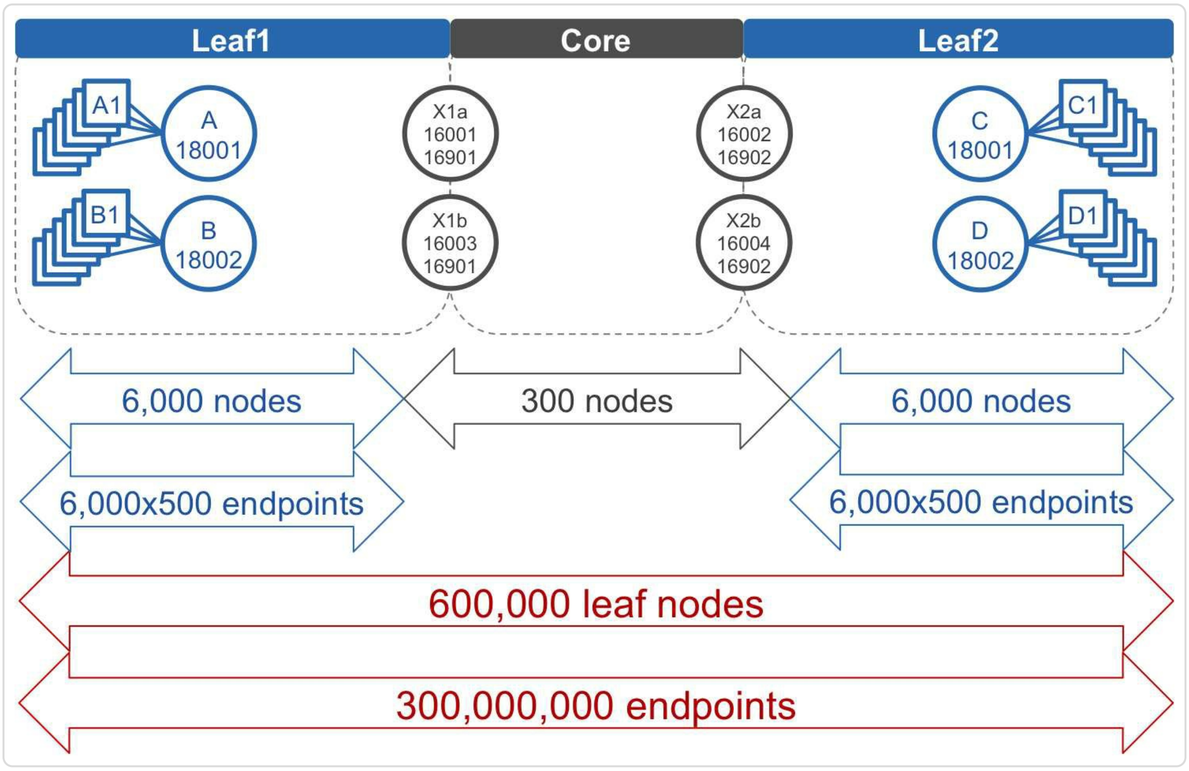


Figure 10-8: Large interconnect scale example

Each X node connects to only one leaf domain, and each leaf domain is connected to the core domain via two X nodes. An Anycast-SID is allocated for each pair of X nodes. Aside of the X nodes, the core domain contains 100 internal nodes to interconnect the X nodes. Assume that a Prefix-SID is assigned to each of these internal core domain nodes. The core domain then has 2 * 100 Prefix-SIDs and 100 Anycast-SIDs for the X nodes, and 100 Prefix-SIDs for the internal core nodes.

All X node Prefix-SIDs are redistributed in each leaf domain.

Network-wide scale:

- 6000 (nodes per leaf domain) * 100 (number of leaf domains) = 600000 nodes
- 6000 (nodes per leaf domain) * 100 (number of leaf domains) * 500

(endpoints per leaf node) = 300 million endpoints

Per-node SID scale:

- Leaf node: 6000 (leaf node Prefix-SIDs) + 200 (X node Prefix-SIDs) + 100 (X node Anycast-SIDs) + 500 (endpoint Adj-SIDs) = 6800 SIDs
- X node: 6000 (leaf node Prefix-SIDs) + 200 (X node Prefix-SIDs) + 100 (X node Anycast-SIDs) + 100 (internal core node Prefix-SIDs) = 6400 SIDs
- Internal core node: 200 (X node Prefix-SIDs) + 100 (X node Anycast-SIDs) + 100 (internal core node Prefix-SIDs) = 400 SIDs

The above calculation does not include the Adj-SIDs of the interconnecting links. These are local to each node and their number is typically < 100.

Depending on the forwarding table capabilities of the leaf nodes, the leaf domains can be split in multiple smaller sub-leaf domains. For example, all leaf domains are reduced to contain 1000 nodes per domain. To maintain the total number of endpoints (300 million), the number of leaf domains is increased to 600. Each X node connects to only one leaf domain, and each leaf domain is connected to the core via two X nodes. Therefore the number of X nodes is 1200. Three Prefix-SIDs are allocated per X node pair: one Prefix-SID for each individual node and one Anycast-SID for the pair. The per-node SID scale then becomes:

- Leaf node: 1000 (leaf node Prefix-SIDs) + 1200 (X node Prefix-SIDs) + 600 (X node Anycast-SIDs) + 500 (endpoint Adj-SIDs) = 3300 SIDs
- X node: 1000 (leaf node Prefix-SIDs) + 1200 (X node Prefix-SIDs) + 600 (X node Anycast-SIDs) + 100 (internal node Prefix-SIDs) = 2900

SIDs

- Internal core node: 1200 (X node Prefix-SIDs) + 600 (X node Anycast-SIDs) + 100 (internal node Prefix-SIDs) = 1900 SIDs

10.5 Deployment Models

This design model can be deployed in a greenfield project, using Segment Routing end-to-end. It can also be deployed in a brownfield project by inter-operating with an existing *seamless MPLS* [draft-ietf-mpls-seamless-mpls] network (seamless MPLS is also called "unified MPLS"), see Figure 10-9.

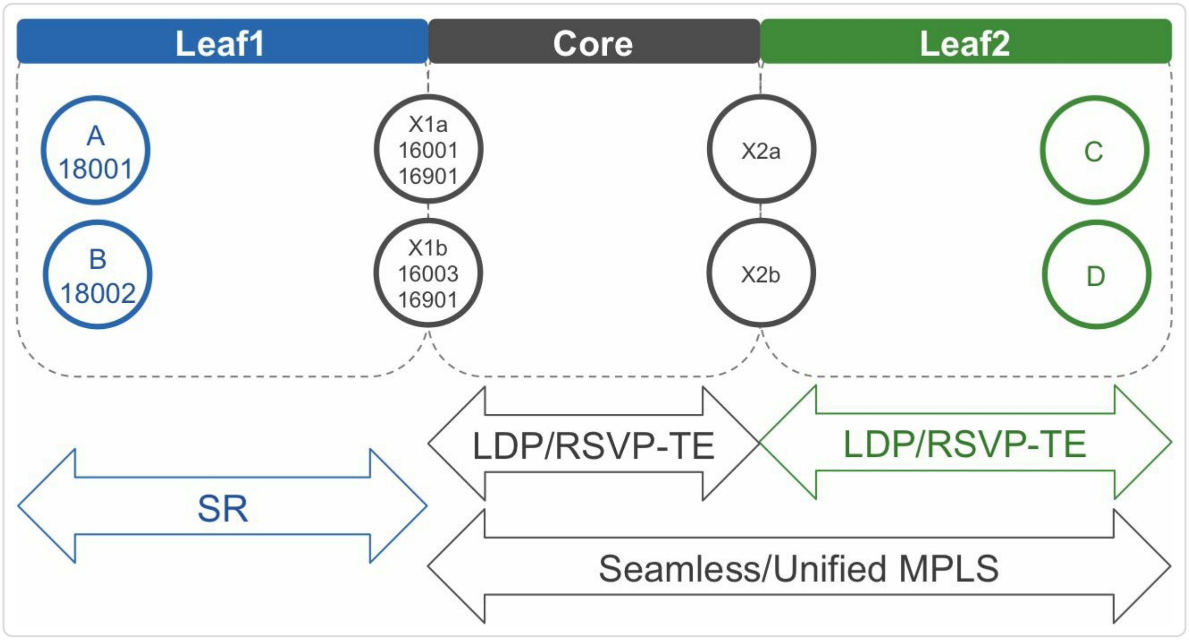


Figure 10-9: Brownfield deployment, interworking with seamless MPLS

One other approach that is widely discussed, is to enable SR in the IGP domains in an existing seamless MPLS network design. The best effort or existing services continue to operate based on the end-to-end connectivity provided via BGP-LU in the hierarchical design (seamless MPLS). New services that leverage SRTE capabilities (to be covered in the next book) are deployed using end-to-end Segment Lists as described in this chapter. The best effort services may later be also migrated over to use similar end-to-end “best effort” Segment Lists. Other design combinations are also possible, but they are beyond the scope of this book.

10.6 Benefits

This design model brings a number of benefits to the large-scale network deployment. It provides a simple way to scale MPLS network using existing SR, no protocol changes are required to support this. The network nodes only need to run a single protocol, an IGP. As each end-to-end path is expressed as a Segment List, they use all available ECMP on the path to the destination for each of the Prefix Segments in the list. Using Anycast Segments for the X nodes (border nodes) enables ECMP to cross the domain boundaries. Topology Independent LFA, which is available once Segment Routing is enabled, provides sub-50 ms protection via local repair for any link/node/SRLG failures for all services, in a simple, automated, scalable and distributed manner There is no operational complexity for implementing different protection mechanisms for different services at the required massive scale across domains. In addition to all this, there is the capability to deliver end-to-end Traffic Engineering capabilities for enabling newer services by the imposition of a Segment List in the packet by the source, without the need to add any per service/flow state anywhere in the network.

"The practice of partitioning of networks into separate IGP domains is becoming more prevalent as they scale and converge to handle multiple services. The reasons for such multi-domain networks are generally scalability aspects, their function (e.g. core, regional aggregation, etc.), their services (e.g. Internet service, VPN service, etc.) or administrative reasons or a mix of these. The large scale interconnect design solution with SR enables these transition and the scaling up of the network. Most importantly it enables the end-to-end multi-domain Traffic Engineering solutions that were not available in a simple and scalable model until now."

— Ketan Talaulikar

10.7 Summary

- Use Segment Routing to scale the network to support hundreds of thousands of network nodes, and tens of millions of underlay endpoints.
- The large scale interconnect model provides a simple to operate flexible network design, e.g. flexible size of the leaf domain, sub-leaf domains, …
- The solution provides inter-operability with existing network designs, e.g. LDP/RSVP-TE, seamless MPLS design.
- The design fully leverages the distributed SR in each domain and provides optimized forwarding and High Availability: leverage ECMP, TI-LFA protection, and providing end-to-end TE capabilities.

10.8 References

- [draft-filsfils-spring-large-scale-interconnect] Filsfils, C., Cai, D., Previdi, S., Henderickx, W., Shakir, R., Cooper, D., Ferguson, F., Lin, S., Laberge, T., Decraene, B., Jalil, L., and J. Tantsura, “Interconnecting Millions Of Endpoints With Segment Routing”, draft-filsfils-spring-large-scale-interconnect (work in progress), September 2016, https://datatracker.ietf.org/doc/draft-filsfils-spring-large-scale-interconnect.
- [draft-ietf-mpls-seamless-mpls] Leymann, N., Decraene, B., Filsfils, C., Konstantynowicz, M., and D. Steinberg, “Seamless MPLS Architecture”, draft-ietf-mpls-seamless-mpls (work in progress), October 2015, https://datatracker.ietf.org/doc/draft-ietf-mpls-seamless-mpls.

11 VERIFYING CONNECTIVITY IN SR MPLS NETWORK

An operator has multiple options to verify reachability to a destination: a ping is often tried first to verify basic connectivity followed by a traceroute to the destination to isolate the point of the failure if ping fails. Traceroute is also used by itself to check whether the actual path that is followed through the destination in the network is as expected by the design; thus it is more than just a failure analysis tool.

This book will often refer to these popular tools as “IP ping” and “IP traceroute” to distinguish them from their MPLS counterparts. IP ping and IP traceroute use Internet Control Message Protocol (ICMP) for their probe and/or return packets. Despite of the name given them, IP ping and IP traceroute can also be used in MPLS networks. ICMP has been extended to improve support of IP traceroute in MPLS networks, but basically they are the same tools.

However, there are some deficiencies of IP ping and IP traceroute when used for verification in MPLS networks, MPLS specific tools are introduced that are based on the ping and traceroute paradigm: MPLS ping and MPLS traceroute, also called LSP ping and LSP traceroute. These are new tools that only have their names in common with their IP counterparts. These MPLS tools do not use ICMP for their probe and return packets, but their own protocol.

This chapter discusses all these tools in more detail and their application and usage in Segment Routing networks

11.1 Existing IP Toolkit

Two classic networking tools – ping and traceroute – are the basis of nearly every network troubleshooting process. Ping verifies connectivity to a destination and with traceroute one can find the point where the network path is broken.

The IP ping and traceroute commands are VRF aware in Cisco IOS XR. They can be used inside a VRF on the L3VPN PE node for troubleshooting purposes and to verify reachability to the CE nodes and other devices in the VPN.

11.1.1 IP Ping

IP Ping is a widely deployed tool. Ping uses ICMP Echo Request and Reply packets to verify connectivity to a destination. It also measures a coarse round trip delay to the destination and can measure coarse packet loss.

The originator sends an ICMP Echo Request packet, which contains an Identifier and a Sequence Number. These two fields are used to match the reply to the request. To measure the round trip time, a timestamp is collected when transmitting the Echo Request. Typically, this timestamp is included in the payload to reduce state that must be kept. The IP TTL of the packet is typically set to 255. The source address is a local reachable address, usually the address of the outgoing interface.

The target responds with an ICMP Echo Reply packet, returning the data received in the Echo Request message. This data includes the Identifier and Sequence Number fields as well as the payload, which typically contains the transmit timestamp.

The originating node receives the ICMP Echo Reply, collects the current timestamp and uses the transmit timestamp, typically embedded in the ICMP Echo Reply packet’s payload, to compute the round trip time of the packet. The result is then displayed to the user.

11.1.2 IP Traceroute

When using the traceroute command, probes are sent to the destination address that the user specified. Usually the probes are UDP packets with a destination UDP port number starting at 33434[1]. The probes are successively sent with incrementing IP TTL field values. The first probe packet has IP TTL set to 1, the next probe packet has IP TTL set to 2, and so on. The UDP destination port number is usually incremented for every probe that is sent. Other implementations may only increment the UDP port when incrementing the TTL value. Nodes in the network can include the UDP destination port in the load-balancing hash computation (L4 or 7-tuple hashing). So every time a different UDP port is used for a probe packet, it may follow another path through the network. This will be visible in the traceroute output.

Since the first probe packet has an IP TTL of 1, the TTL of the packet expires on the nexthop node. That node generates an Internet Control Message Protocol (ICMP) message of type “Time Exceeded” (ICMP type 11, code 0) and sends that to the source address of the probe packet. This ICMP message also includes the header and part of the payload of the probe packet. The source address of this ICMP message is the IP address of the interface on which it received the probe packet.[2]

The traceroute program can correlate the received ICMP message to the probe packet based on the UDP port of the probe packet, since the ICMP message also includes the original header of the probe packet.

The output of the traceroute command shows the TTL value of the probe packet, the source address of the returned ICMP message and the round trip delay. The source address of the ICMP message is the ingress interface of the probe packet on the node that generated this ICMP packet. By default, three probes are sent for each TTL value to get more delay statistics for every hop on the path and to compensate for possible packet loss.

Next, the traceroute program sends a probe packet with IP TTL set to 2. The nexthop node forwards this packet as usual, it decrements the TTL as for any packet. The probe packet's TTL expires on the second node on the path to the destination. This node sends an ICMP "Time Exceeded" message and the traceroute output displays the information of the second hop.

This process is repeated until the destination specified in the traceroute command is reached without the TTL expiring. The destination node is (hopefully) not expecting any packets on the UDP destination port of the probe packet, there is no application using that UDP port. The reason to use the UDP destination ports starting at 33434, is to make it unlikely that an application is using that UDP port. The Internet Assigned Numbers Authority (IANA) has allocated UDP port 33434 to traceroute, and it has left ports 33435-33655 unallocated. Because the destination node is not listening to the UDP port, it generates an ICMP "Port Unreachable" (ICMP type 3, code 3) message and sends it to the source address of the probe packet. The output of the traceroute command then displays the information of the last hop. Here the traceroute command stops.

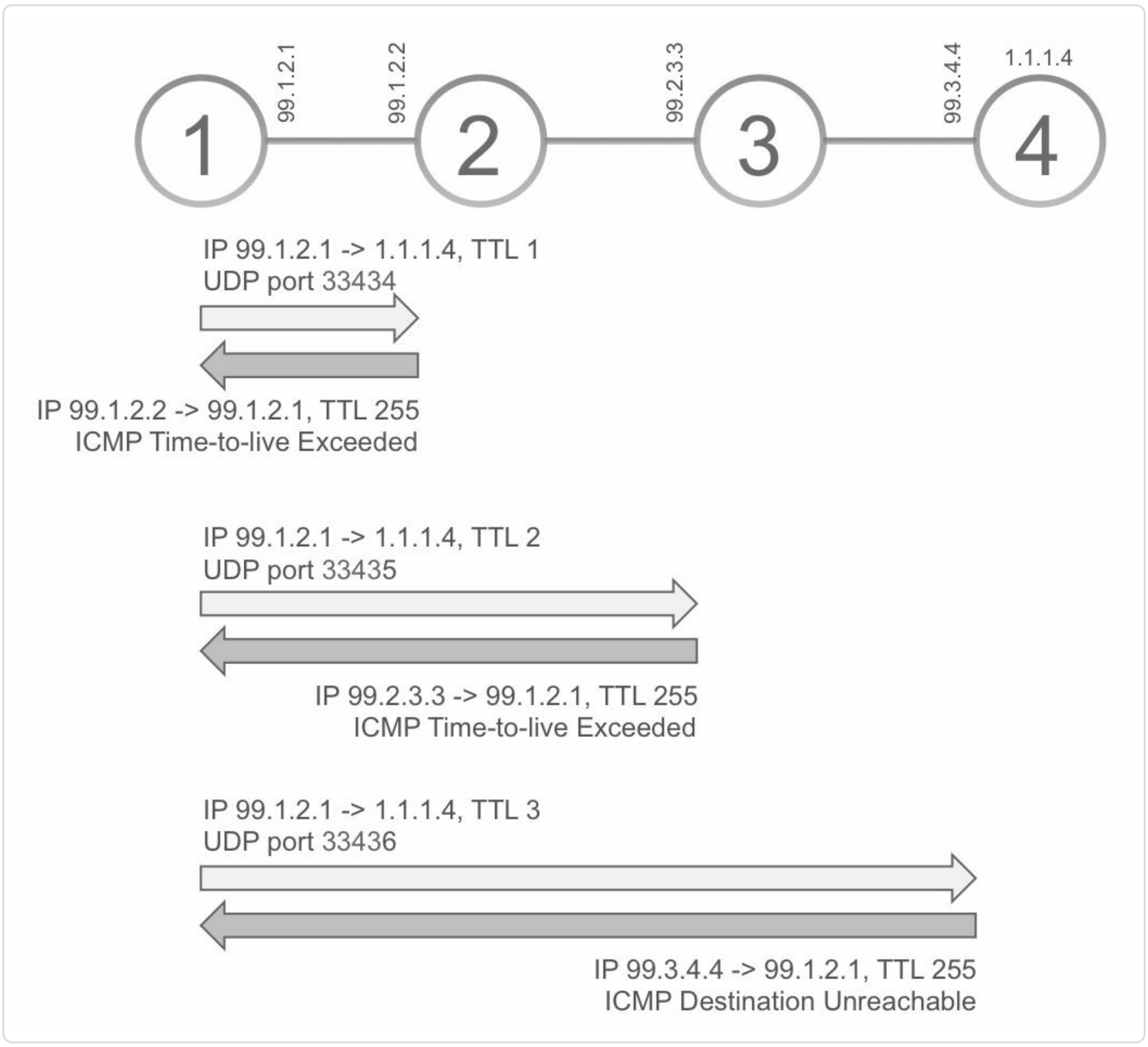

99.1.2.1
99.1.2.2
99.2.3.3
99.3.4.4
1.1.1.4
1
2
3
4
IP 99.1.2.1 -> 1.1.1.4, TTL 1
UDP port 33434
IP 99.1.2.2 -> 99.1.2.1, TTL 255
ICMP Time-to-live Exceeded
IP 99.1.2.1 -> 1.1.1.4, TTL 2
UDP port 33435
IP 99.2.3.3 -> 99.1.2.1, TTL 255
ICMP Time-to-live Exceeded
IP 99.1.2.1 -> 1.1.1.4, TTL 3
UDP port 33436
IP 99.3.4.4 -> 99.1.2.1, TTL 255
ICMP Destination Unreachable

Figure 11-1: Example IP traceroute in IP network

11.1.3 IP Ping, Traceroute and ECMP

Equal Cost Multi-Paths are omnipresent in IP networks. In the presence of ECMP, the network load-balances traffic over the different paths to reach the destination. A failure in one of these equal cost paths results in partial packet loss and may stay undetected for an ECMP-unaware path verification mechanism.

IP ping and traceroute are not inherently ECMP-aware; the path that the probe packets use to reach their destination depends entirely on the load-balancing hashing mechanism of the transit nodes. The operator could modify the IP header fields of the probe packets to try steering them on the desired path. The operator could for example specify a different source address in the ping or traceroute command. However, this is a cumbersome method, to say the least.

IP ping and traceroute do not support path discovery by design. They do not provide an exhaustive and deterministic methodology to discover all ECMP towards a destination in the network. Also they do not provide an easy way to exercise a specific path of the available ECMP, since they are restricted to using reachable IP source and destination addresses for the probe packets.

11.1.4 IP Ping in MPLS Environment

The IP ping tool does not know about the MPLS network. It sends its probe packets as ICMP packets and lets the routing layer decide on how these packets are forwarded. If IP ping is used in an MPLS network, the routing layer imposes the same labels over the generated ICMP packets as on other IP packets going to the same destination. Also the ICMP Echo Reply packets can follow the MPLS return path if the routing layer decides so. So, basic functionality of IP ping works in an MPLS network. The biggest disadvantage of IP ping in an MPLS environment is that it cannot detect LSP failures that do not affect IP connectivity. Ping returns success as long as there is IP reachability, even if the Label Switched Path is broken. Assume for example that the label switched path between two PEs is broken. While troubleshooting, the operator tries a ping between the two PEs. This ping is still successful since the network continues forwarding

ICMP packets using regular IP forwarding, as unlabeled packets. It looks as if the end-to-end connectivity is preserved, while the label switched path is actually broken. This situation results in service traffic being dropped, irrespective the positive ping result.

11.1.5 IP Traceroute in MPLS Environment

IP traceroute can also be used in an MPLS environment. But same as for IP ping, there are limitations with using the IP traceroute command in such environment. The IP traceroute tool does not know about the MPLS network. It sends its probe packets as IP packets and lets the routing layer decide on how these packets are forwarded. If they get an MPLS header, the IP TTL value is copied to the MPLS TTL field. The MPLS TTL field of the top label is decremented when the packet travels through the network.

If the TTL of the top label of the MPLS encapsulated probe packet expires on a node, the node generates an ICMP “Time Exceeded” message. As described in chapter 3, “Segment Routing MPLS Data Plane”, the node adds the full MPLS label stack of the received probe packet in the ICMP message, using the ICMP extensions (IETF RFC 4950). The node then imposes the label stack of the received probe packet on the generated ICMP message, as specified in IETF RFC 3032. Therefore the ICMP message is then forwarded downstream on the original LSP of the probe packet. If the Label Switched Path (LSP) to the destination is intact, then the ICMP message reaches the end of the LSP. At that point the IP header of the ICMP packet is exposed and from there this ICMP packet is forwarded to its destination address using regular forwarding. The destination of this ICMP message is the node running the traceroute command.

The traceroute program receives the ICMP message and displays the information to the user, including the MPLS label stack that was embedded in the ICMP message. That label stack is the label stack as received by the node where the TTL expired.

The traceroute program also presents a delay for each hop. To compute this the node collects a transmit timestamp when the probe packet is sent and collects a receive timestamp when it receives the returning ICMP message. Many implementations encode the transmit timestamp in the probe packet payload to avoid having to keep state for every probe packet. The difference between the two timestamps is presented to the user in the output. This means that only the round-trip delay is measured; nodes along the path are not involved in the delay measurement. So, traceroute shows the hops along the forward path, but the delay shown for each hop is the round-trip delay.

Figure 11-1 and Example 11-1 illustrate the usage of IP traceroute in an MPLS network. The network topology consists of a chain of four nodes, with SR enabled on all nodes. Node4 advertises its loopback prefix 1.1.1.4/32 with a Prefix-SID 16004. A traceroute is executed on Node1 for destination 1.1.1.4. A single probe is sent for each TTL value (`probe 1` option in command). The first probe is sent with IP TTL 1, this TTL is copied to the TTL field of the imposed label 16004. The probe packet arrives on Node2 with label 16004 and TTL 1. Node2 decrements the TTL, which is now zero. Therefore, Node2 generates an ICMP Time-to-live Exceeded message towards the source address 99.1.2.1 of the received packet. The source address of the ICMP packet is the IP address of the interface on which the probe packet was received. Node2 imposes the label 16004 of the received probe packet on the ICMP packet and sends it to the nexthop for label 16004. This packet is forwarded as a regular packet. It eventually arrives on Node4 without label and Node4 forwards the packet based on the IP destination address. Finally it arrives on Node1, where the traceroute process produces the output for the user. Traceroute shows the TTL of the originated probe packet, the source address of the ICMP message and the label stack included in the ICMP message.

Example 11-1: IP traceroute in MPLS network – traceroute output

```
RP/0/0/CPU0:xrvr-1#traceroute 1.1.1.4 probe 1

Type escape sequence to abort.
Tracing the route to 1.1.1.4

 1  99.1.2.2 [MPLS: Label 16004 Exp 0] 0 msec
 2  99.2.3.3 [MPLS: Label 16004 Exp 0] 0 msec
 3  99.3.4.4 0 msec
```

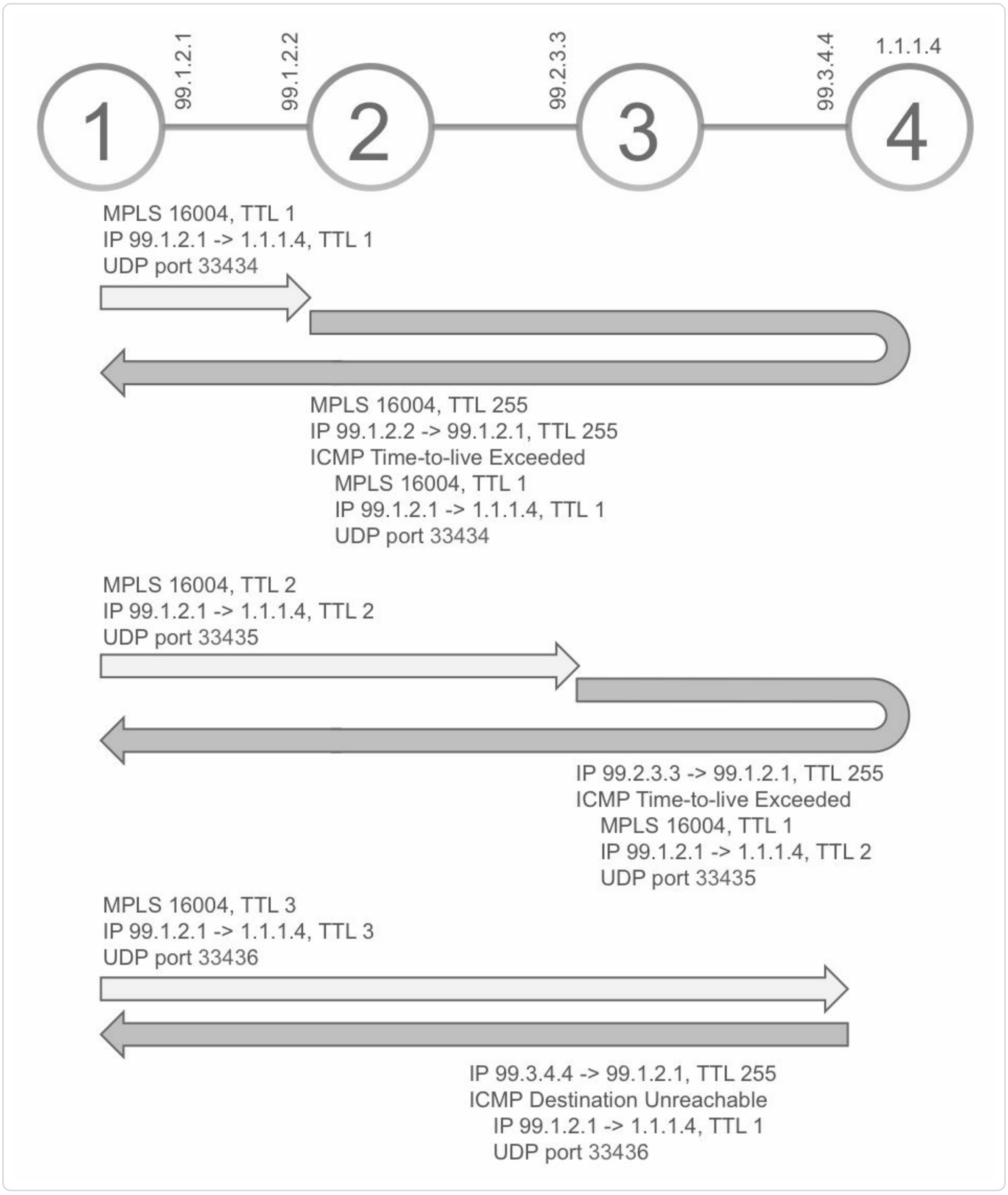

99.1.2.1
99.1.2.2
99.2.3.3
99.3.4.4
1.1.1.4
1
2
3
4
MPLS 16004, TTL 1
IP 99.1.2.1 -> 1.1.1.4, TTL 1
UDP port 33434
MPLS 16004, TTL 255
IP 99.1.2.2 -> 99.1.2.1, TTL 255
ICMP Time-to-live Exceeded
MPLS 16004, TTL 1
IP 99.1.2.1 -> 1.1.1.4, TTL 1
UDP port 33434
MPLS 16004, TTL 2
IP 99.1.2.1 -> 1.1.1.4, TTL 2
UDP port 33435
IP 99.2.3.3 -> 99.1.2.1, TTL 255
ICMP Time-to-live Exceeded
MPLS 16004, TTL 1
IP 99.1.2.1 -> 1.1.1.4, TTL 2
UDP port 33435
MPLS 16004, TTL 3
IP 99.1.2.1 -> 1.1.1.4, TTL 3
UDP port 33436
IP 99.3.4.4 -> 99.1.2.1, TTL 255
ICMP Destination Unreachable
IP 99.1.2.1 -> 1.1.1.4, TTL 1
UDP port 33436

Figure 11-2: IP traceroute in MPLS network

Using IP traceroute has limitations when used in an MPLS environment. It has the same issue as the IP ping utility. If the MPLS LSP is broken, but the IP forwarding is still functional (e.g. outgoing label "Unlabelled"), then the node next to the failure tries to forward the probe packet using regular IP forwarding if it has a single label. The node removes the one label and forwards the packet as an IP packet to the destination. The probe packet may then reach the destination and the response will return to the source. Since a return packet was received, the broken MPLS LSP may stay unnoticed. The MPLS LSP breakage may still be discovered by watching the MPLS label stack information in the returning ICMP message. If Node3 has no SR (and no LDP) enabled, then the LSP to Node4 is broken. Executing a traceroute from Node1 to 1.1.1.4 then results in the output of Figure 11-2. The ping to 1.1.1.4 is successful, as well as the traceroute. Notice the lack of label information for the second hop in the traceroute (line 13). Since Node3 is not SR enabled, Node2 forwarded the probe packet to Node3 as an unlabeled IP packet.

Example 11-2: Traceroute over faulty LSP

```
 1| RP/0/0/CPU0:xrvr-1#ping 1.1.1.4
 2| Type escape sequence to abort.
 3| Sending 5, 100-byte ICMP Echos to 1.1.1.4, timeout is 2
seconds:
 4| !!!!!
 5| Success rate is 100 percent (5/5), round-trip min/avg/max =
1/4/9 ms
 6|
 7| RP/0/0/CPU0:xrvr-1#traceroute 1.1.1.4 probe 1
 8|
 9| Type escape sequence to abort.
10| Tracing the route to 1.1.1.4
11|
12|  1  99.1.2.2 [MPLS: Label 16004 Exp 0] 9 msec
13|  2  99.2.3.3 0 msec
14|  3  99.3.4.4 9 msec
```

Another problem shows up when using IP traceroute to try locating the point where the LSP is broken. Since the generated ICMP messages first travel to the end of the original LSP before returning to the originator, *any* failure in that LSP may prevent the ICMP messages to return. Localization of the breaking point of the LSP is not possible since *none* of the probe packets gets a response. This problem will show up if the probe packet has more than one label (for example when using traceroute within a VRF) or if the core nodes do not know the IP destination address of the probe packet (for example when using traceroute between PEs in a BGP-free core).

Sometimes the traceroute tool is used to coarsely verify packet delays between hops on the path. Remember that traceroute only calculates the round-trip delay for each hop along the path.Now, if every returning ICMP message first travels all the way to the end of the LSP, then the calculated delay for each hop along this LSP is in fact the round-trip delay to the tail node of the LSP. No information is provided for the delay between the hops on the LSP. By default Cisco IOS XR uses this ICMP tunneling mechanism (IETF RFC 3032); a node that generates an ICMP message as response to a labeled packet first steers the ICMP message to the tail of the LSP. This is the case even if the node has reachability to the node that originated the probe and could have returned the ICMP message directly to the originating node. This default behavior can be changed with the`mpls ipv4 ttl-expiration-pop <n>` configuration. This configuration specifies that the ICMP tunneling mechanism must only be applied for ICMP messages in response to packets with more than *n* labels. For packets with *n* or less labels, the ICMP message must be sent directly to the source of the probe packet. This mitigates some of the drawbacks of IP traceroute in an MPLS environment.

HIGHLIGHT: IP ping and traceroute

Provide basic connectivity checks, error isolation and path verification abilities and are widely supported and used all over networks

Do not support discovery of ECMP paths nor guarantee that all ECMP paths would get verified; this depends on the implementation of these tools and how routers do load balancing

Work with both IP and MPLS forwarding paths hence also for Segment Routing

In Segment Routing deployment, traceroute shows the labels used for segments over the path to the destination; same global segment with homogenous SRGB and also help verify the label cross connects at points of SR/LDP interworking

Can be used to measure coarse packet loss and round trip time

11.2 Existing MPLS Toolkit

IP ping and IP traceroute are often used to diagnose the cause of a reachability problem in the network. These tools *can* be used in a MPLS environment, but tracing failures can be difficult if there are inconsistencies between IP and MPLS forwarding tables, inconsistencies between MPLS control plane and forwarding table, or MPLS traffic blackholing. Many of these MPLS problems are difficult or impossible to diagnose with IP tools.

IETF RFC 4379 specifies MPLS tools that have been specifically designed to diagnose MPLS transport problems. Modeled after the classic ping and traceroute tools, the MPLS toolkit provides a ping functionality to verify connectivity and a traceroute functionality to verify the path hop-by-hop to locate a fault.

MPLS Ping and Traceroute overcome several limitations of the IP Ping and Traceroute tools:

- The MPLS tools verify the MPLS data path by ensuring the probe packets do not get IP routed.
- The payload of the probe packets carries relevant information for verification.
- The MPLS tools perform MPLS consistency checks by design:
    - Query the control plane to build a probe packet on the source node and verify the forwarding and control plane at egress.
    - The probe packet traverses the MPLS data path and the node that processes the probe verifies if forwarding is consistent.

- The MPLS tools are extensible by design using TLV mechanism.
- The toolkit supports path discovery by design.
- The probe packets contain timestamps to allow measuring coarse one-way packet delays.

11.2.1 MPLS ping

Contrary to their IP counterparts, the MPLS tools do not use ICMP, but rely on IPv4 or IPv6 UDP packets. The base of the MPLS Ping and Traceroute tools (also called “LSP Ping” and “LSP Traceroute”) are the MPLS Echo Request and MPLS Echo Reply packets.

To validate a Label Switched Path, the MPLS Echo Request packet must be sent in-band of this LSP, they must follow the same path as other packets carried by the LSP. Therefore, the same label stack is applied to the MPLS Echo Request packets as for other packets carried by this LSP. MPLS Echo Request packets mainly validate the MPLS data plane, but they can also be used to check for inconsistencies between data plane and control plane. Therefore the MPLS Echo packets carry additional information.

MPLS Echo Reply packets can be handled in different ways, but commonly they are returned to the originator of the MPLS Echo Request packet as an IP packet, following the regular forwarding in the network. The MPLS Echo Reply packet can thus take an MPLS or an IP path to return to the originating node. In such case, the LSP is only validated in one direction, the return LSP is not validated. But the MPLS Echo Reply can be dropped on the return path, which may lead to a false negative.

An MPLS Echo Request packet is an UDP packet (with UDP destination port 3503) that is sent to a target node by using the label stack of the LSP that is being validated. The usage of the label stack is significant as it forwards the packet in-band of the LSP.

Precautions are taken to ensure that the packet cannot deviate from the intended path unnoticed:

1. The IP destination address of an MPLS Echo Request packet is selected from the IPv4 127.0.0.0/8 range or the equivalent IPv4-mapped IPv6 range ::ffff:127.0.0.0/104 for IPv6. This address range is the so-called host loopback address range. Packets destined to any destination in 127.0.0.0/8 should never appear on any network. Using such address as IP destination address forces the packet to be consumed by the node receiving it. In the case of the MPLS Echo Request, that is either the ultimate node of the LSP or another node if for some reason the label stack would be erroneously removed from the packet. Since the IP destination address is not actually used for forwarding, it can also be varied in order to exercise different paths in an ECMP. As explained in section chapter 3, “Segment Routing MPLS Data Plane”, packets are load-balanced based on a hash computation, and the IP destination address is an element of that hash computation.
2. The IP TTL field of the packet is set to 1. Note that this does not apply to the MPLS TTL field.
3. The Router Alert option is set in the IP header.

The MPLS Echo Request packet carries additional information about the LSP to enable the destination node, the tail-end of the LSP, to verify if it is the intended target of the probe. Therefore the source node includes the Forwarding Equivalence Class (FEC) information of the LSP in the generated MPLS Echo Request packet. The tail-end node of the LSP verifies if it is indeed the tail-end (also called "egress") node for the FEC specified in the probe packet. If that is the case, then it returns "success" in the MPLS Echo Reply message, otherwise it returns an error code.

An MPLS Echo Reply packet is sent in response to an MPLS Echo Request packet by the node that processes it. In order to originate or respond to the MPLS Echo packets in Cisco IOS XR, `mpls oam` must be configured. Each node that receives an MPLS Echo Request packet without an MPLS header processes the packet, if `mpls oam` is configured. This node can be the destination node or an intermediate node if the LSP is broken and the IP header is prematurely exposed. Also a node that receives an MPLS Echo Request packet with an MPLS TTL = 1 processes the packet.

Example 11-3 shows the console output of an MPLS ping on Node1 to 1.1.1.4/32. Contrary to IP ping, a *prefix* 1.1.1.4/32 is specified instead of an address. A generic FEC is included in the MPLS Echo Request packet. More details of the FEC types are discussed after this example. The network topology used in this example is displayed in Figure 11-3.

Example 11-3: Example output of MPLS ping

```
RP/0/0/CPU0:xrvr-1#ping mpls ipv4 1.1.1.4/32 fec generic

Sending 5, 100-byte MPLS Echos to 1.1.1.4/32,
      timeout is 2 seconds, send interval is 0 msec:
```

```
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output
interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC
mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx
label,
  'P' - no rx intf label prot, 'p' - premature termination of
LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max =
10/12/20 ms

image::0087_SR_book_SR_MPLS_OAM_v5p_files/image11-
3.png[caption='Figure 11-3: ', title='MPLS Ping example']
```

Example 11-4 shows a packet capture of an MPLS Echo Request packet. The packet is captured on the link between Node1 and Node2. Notice the MPLS label 16004 in line 1. This is the Prefix-SID label for 1.1.1.4/32, the loopback prefix of Node4. The IP TTL (`ttl 1`) and IP Router Alter option (`options (RA)`) are shown in line 2. The packet is a UDP packet with IP destination address 127.0.0.1 and UDP port 3503, as displayed in line 3. The target FEC is shown in lines 14 to 17; it is the prefix 1.1.1.4/32. The other elements in the payload are discusses further in this section.

Example 11-4: Example MPLS Echo Request packet

```
1| 17:37:18.286050 MPLS (label 16004, exp 0, [S], ttl 255)
2|         IP (tos 0x0, ttl 1, id 78, offset 0, flags [DF],
proto UDP (17), length 96, options (RA))
3|     99.1.2.1.3503 > 127.0.0.1.3503: [udp sum ok]
4|         LSP-PINGv1, msg-type: MPLS Echo Request (1), length:
64
```

```
            Global Flags: 0x0000
            reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
            Return Code: No return code or return code
contained in the Error Code TLV (0)
            Return Subcode: (0)
            Sender Handle: 0x0000619f, Sequence: 5
            Sender Timestamp: 17:36:52.371585 Receiver
Timestamp: no timestamp
            Vendor Private Code TLV (64512), length: 12
              Vendor Id: ciscoSystems (9)
              Value: 0001000400000004
            Target FEC Stack TLV (1), length: 12
              Generic IPv4 prefix subTLV (14), length: 5
              IPv4 Prefix: 1.1.1.4 (1.1.1.4)
              Prefix Length: 32
```

Example 11-5 shows a captured MPLS Echo Reply packet that was sent in response to the Echo Request above. This packet is captured on the link between Node1 and Node2. This MPLS Echo Reply packet is an IP packet with destination address 99.1.2.1, the source IP address of the Echo Request packet. The other elements in the payload are discusses further in this section.

Example 11-5: Example MPLS Echo Reply packet

```
17:37:18.294067 IP (tos 0xc0, ttl 253, id 62, offset 0, flags
[none], proto UDP (17), length 76)
    99.3.4.4.3503 > 99.1.2.1.3503: [udp sum ok]
        LSP-PINGv1, msg-type: MPLS Echo Reply (2), length: 48
          Global Flags: 0x0000
          reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
          Return Code: Replying router is an egress for the FEC
at stack depth 1 (3)
          Return Subcode: (1)
          Sender Handle: 0x0000619f, Sequence: 5
          Sender Timestamp: 17:36:52.371585 Receiver Timestamp:
17:36:53.263483
          Vendor Private Code TLV (64512), length: 12
            Vendor Id: ciscoSystems (9)
            Value: 0001000400000004
```

11.2.2 MPLS Echo Request/Reply Packet

The format of a MPLS Echo Request and MPLS Echo Reply packet payload as specified in IETF RFC 4379 is shown in Figure 11-4.

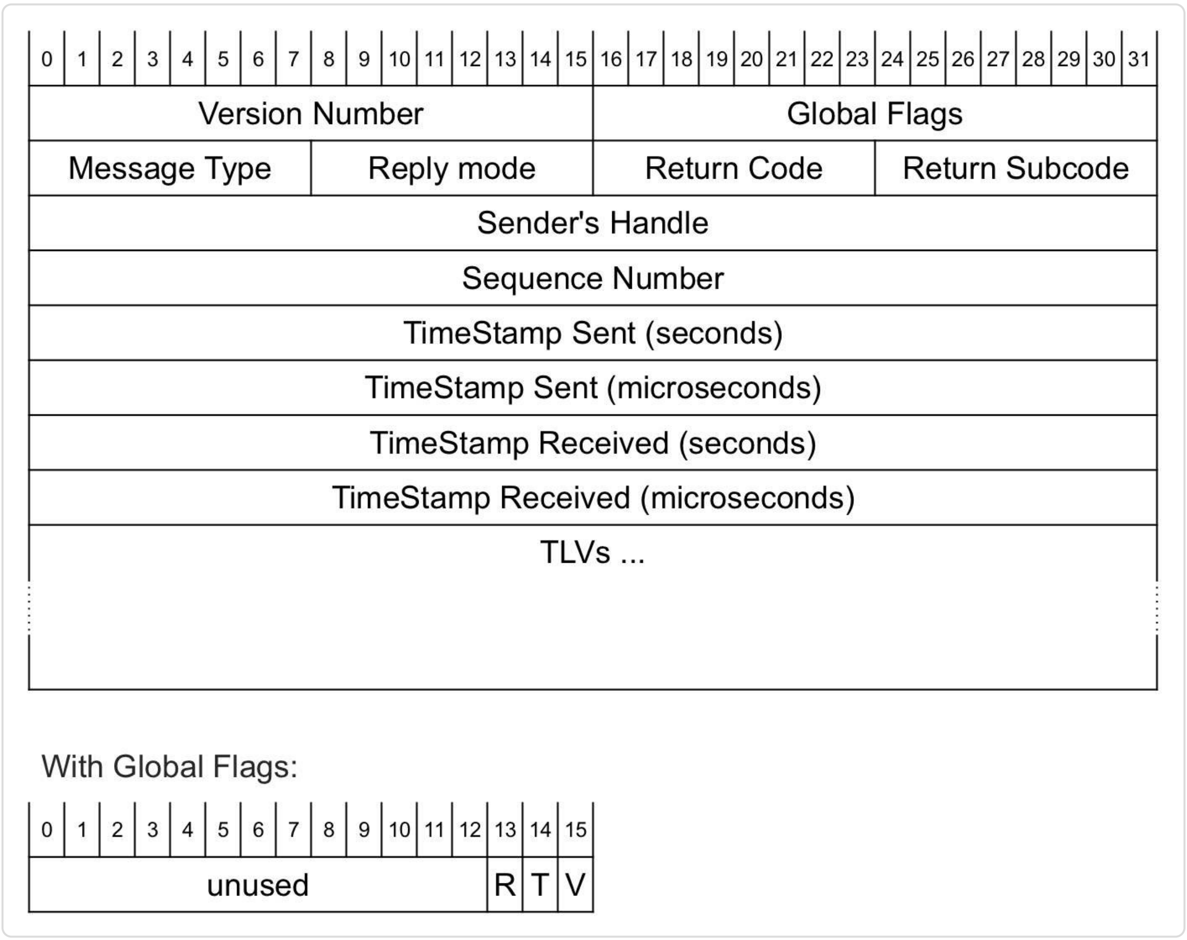


Figure 11-4: Format of MPLS Echo Request/Reply message payload

- Version number: 1
- Global flags:
  - V-flag (Validate FEC stack): set if the sender requests validation of the FEC stack; unset if the sender leaves the choice to validate FEC to the receiver.
  - T-flag (respond only if TTL expired): if set, then an Echo Reply must only be sent when the TTL of the packet has expired – IETF

RFC 6425

    - R-flag (validate Reverse path): if set, then the responder should return reverse-path FEC information – IETF RFC 6426

- Message type: Echo Request or Echo Reply (and other that are not described in this book)
- Reply mode: describes how the Echo Reply should be returned; in Cisco IOS XR the Reply is returned as an IPv4/IPv6 UDP packet by default for most LSP types.
- Return code and Return Sub-code: identifies the result of processing the Echo Request
- Sender's handle: An identifier that allows correlating the MPLS Echo Reply to the original MPLS Echo Request.
- Sequence number: used to detect missed replies
- Timestamp sent: time-of-day when the MPLS Echo Request was sent
- Timestamp received: time-of-day when the MPLS Echo Request was received
- TLVs: different TLVs can be included, see the IANA registry.[3] Only a subset of the TLVs is covered in this book.

One of the TLV types that can be included are vendor specific TLVs. These vendor private TLVs have a type value between 31744-32767 or 64512-65535. TLVs with type values larger than 32767 are optional and a node should ignore them if it does not understand it. The Cisco extension TLV was added to ensure interoperability between implementations of different versions of the IETF draft that eventually resulted in IETF RFC 4379. The MPLS Echo packets in the examples include a Cisco extension TLV with a revision sub-TLV that contains the latest revision, represented by an internal revision number 4. See for example Example 11-4, lines 11 to 13.

Since an MPLS Echo Request is used to test a particular LSP, it must include a Target FEC Stack TLV; this TLV specifies the FEC(s) of the intended LSP. This TLV is required such that the destination node can verify if it is indeed the LSP tail-end node for the specified FEC. For example, if a node wants to verify if Prefix-SID 16004 for prefix 1.1.1.4/32 indeed reaches the node that advertised this Prefix-SID, that node can send an MPLS Echo Request with a label 16004 and include a FEC Stack TLV with a single FEC, namely prefix 1.1.1.4/32.

When a node receives a MPLS Echo Request, it validates if both control plane and data plane are in sync and both are in accordance with what is specified in the Target FEC Stack TLV of the received packet before generating an MPLS Echo Reply packet.

An example of an MPLS Echo Request packet is shown in Example 11-4. In this example the Target FEC Stack TLV contains a generic IPv4 Prefix sub-TLV, shown in lines 14 to 17 of the output.

Each type of FEC has its own FEC Type sub-TLV. There are FEC Type sub-TLVs for LDP prefixes, RSVP-TE LSPs, BGP-LU prefixes, VPN prefixes, etc. Each FEC Type sub-TLV contains the necessary fields to specify a FEC of that type.

At the time of writing this book, two FEC Type sub-TLVs are available in Cisco IOS XR to verify SR MPLS paths: “Generic IP Prefix” and “NIL-FEC”. New Segment Routing FEC types and procedures are specified in IETF [draft-ietf-mpls-spring-lsp-ping].

The Generic Prefix FEC type is generally used if the protocol advertising the label is unknown or may vary along the path. The Generic Prefix FEC Type sub-TLV contains the IPv4 or IPv6 prefix and its prefix length.

The NIL-FEC FEC type is generally used if labels from the reserved range, such as the Router Alert label or the Explicit-null label, are added to the label stack. The NIL-FEC FEC Type sub-TLV contains the label value. When using this Nil-FEC Type, Cisco IOS XR adds a single FEC-type sub-TLV in the MPLS Echo Request packet: the IPv4 Explicit-Null label value “0”.[4]

Cisco IOS XR implementation allows to perform the MPLS ping and traceroute operations for IPv4/IPv6 prefix FECs without specification of the FEC type. The implementation will automatically determine the FEC type to be used based on the MPLS control protocol signaling the prefix. However, this approach does not work for Segment Routing prefixes at the time of writing this book. Similar functionality will be extended for Segment Routing prefixes once the implementation of the new Segment Routing extensions are released.

Example 11-6 shows an example of an MPLS ping using NIL-FEC. LSP Ping cannot determine the label stack, outgoing interface, and nexthop when using NIL-FEC, therefore the user needs to specify that information in the traceroute command. The network topology for this example is shown in Figure 11-3.

Example 11-6: console output of MPLS ping with Nil-FEC

```
RP/0/0/CPU0:xrvr-1#ping mpls nil-fec labels 16004 output interface Gi0/0/0/1 nexthop 99.1.2.2

Sending 5, 100-byte MPLS Echos with Nil FEC with labels [16004],
      timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output
interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC
mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no
rx label,
  'P' - no rx intf label prot, 'p' - premature termination
of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max =
1/8/10 ms
```

Example 11-7 shows an MPLS Echo Request packet with a NIL-FEC Target FEC Stack. This packet is captured on the interface between Node1 and Node2 for the MPLS ping of Example 11-6. The label stack of the packet contains two labels: 16004 and 0, see lines 1 and 2. The Target FEC TLV with NIL-FEC sub-TLV is displayed in lines 15 to 17. The label in the NIL-FEC sub-TLV is the IPv4 Explicit-null label (0).

Example 11-7: MPLS Echo Request packet with Nil-FEC

```
13:46:47.892495 MPLS (label 16004, exp 0, ttl 255)
        (label 0 (IPv4 explicit NULL), exp 0, [S], ttl 255)
        IP (tos 0x0, ttl 1, id 106, offset 0, flags [DF], proto
UDP (17), length 92, options (RA))
    99.1.2.1.3503 > 127.0.0.1.3503:
        LSP-PINGv1, msg-type: MPLS Echo Request (1), length: 60
          Global Flags: 0x0000
          reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
          Return Code: No return code or return code contained
in the Error Code TLV (0)
          Return Subcode: (0)
          Sender Handle: 0x000061a7, Sequence: 1
          Sender Timestamp: 13:46:21.177759 Receiver Timestamp:
no timestamp
          Vendor Private Code TLV (64512), length: 12
            Vendor Id: ciscoSystems (9)
            Value: 0001000400000004
          Target FEC Stack TLV (1), length: 8
            Nil FEC subTLV (16), length: 4
              Label: 0 (IPv4 Explicit-Null)
```

The MPLS Echo Reply that is returned by Node4 is shown in Example 11-8. This packet is equivalent to the MPLS Echo Reply packet in Example 11-5.

Example 11-8: MPLS Echo Reply packet for Nil-FEC

```
13:46:47.899147 IP (tos 0xc0, ttl 253, id 5, offset 0, flags
[none], proto UDP (17), length 76)
    99.3.4.4.3503 > 99.1.2.1.3503:
        LSP-PINGv1, msg-type: MPLS Echo Reply (2), length: 48
          Global Flags: 0x0000
          reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
          Return Code: Replying router is an egress for the FEC
at stack depth 1 (3)
          Return Subcode: (1)
          Sender Handle: 0x000061a7, Sequence: 1
          Sender Timestamp: 13:46:21.177759 Receiver Timestamp:
13:46:22.203351
          Vendor Private Code TLV (64512), length: 12
            Vendor Id: ciscoSystems (9)
            Value: 0001000400000004
```

The Target FEC Stack TLV may contain more than one sub-TLV, one for each FEC in the label stack being tested. The first sub-TLV corresponds to the top of the label stack.

11.2.3 MPLS Traceroute

MPLS Traceroute uses basically the same packets and procedures as MPLS Ping. MPLS Traceroute verifies the data plane and control plane at each hop of the LSP to isolate faults. Traceroute sends subsequent MPLS Echo Request packets with incrementing TTL, starting with TTL of 1. This is equivalent to IP traceroute. The transit node where the TTL expires processes the MPLS Echo Request in software and verifies if it has an LSP for the target FEC and if it is the intended transit node for the LSP. The transit node then sends an MPLS Echo Reply containing the result of the verification and, if verification was successful, the outgoing label stack of the LSP, the address of the nexthop node and the nexthop's interface address. The originator processes this MPLS Echo Reply to build the subsequent MPLS Echo Request packet with TTL + 1. This procedure is repeated until the destination replies that it is the tail-end of the LSP.

A Downstream Mapping (DSMAP) TLV is used to carry the outgoing (or downstream) information of the LSP: outgoing label stack, nexthop node and interface addresses.

This TLV can be included in both Echo Request and Echo Reply packets. In the Echo Request packet, it provides the information for the transit node to verify if it is the intended transit node for the LSP and to verify if the packet is received on the intended interface from the upstream node (e.g. when multiple equal cost interfaces connect to the same upstream neighbor). In the Echo Reply packet, it returns the information that the originator will provide to the next node on the LSP. This is yet another fundamental improvement over IP traceroute that enables path discovery and verification as we will see further in this chapter.

IETF RFC 6424 deprecates the DSMAP TLV and replaces it with the Downstream Detailed Mapping (DDMAP) TLV, but the DSMAP TLV is still commonly used, for example in Cisco IOS XR.

The DSMAP TLV format is specified in IETF RFC 4379 and is shown in Figure 11-5.

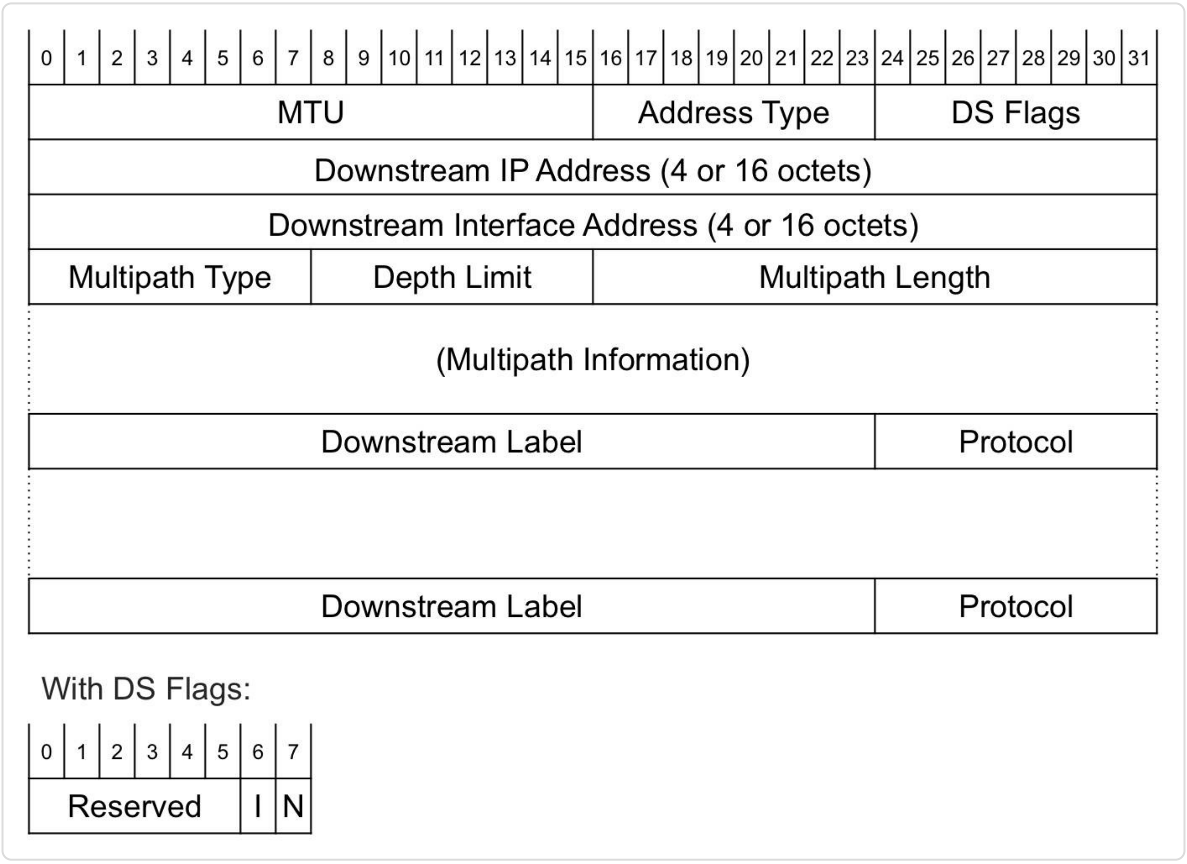


Figure 11-5: Downstream Mapping TLV format

- MTU: maximum size of the MPLS frame (including label stack) that can be sent on the outgoing interface to the downstream neighbor, basically the IP MTU of the downstream interface
- Address Type: IPv4 or IPv6, Numbered or Unnumbered
- DS Flags:
    - I-flag (Interface and Label Stack Object Request): if set, then responding node should reply with Interface and Label Stack Object
    - N-flag (Non-IP packet): the packet flow that is being diagnosed is a

non-IP packet flow, but MPLS Echo packets are IP packets. Therefore the receiving node should treat this MPLS Echo packet as a non-IP packet to properly diagnose the behavior of a non-IP flow.

- Downstream IP Address and Downstream Interface Address: in the Echo Reply, it specifies the downstream neighbor's router ID and interface IP address or interface index, depending on the interface being numbered or unnumbered. In the Echo Request, it contains the information that was provided by the upstream node.
- Multipath Type: indicates if *all* packets will be forwarded to this outgoing interface or only a subset of all packets. For example, if the outgoing interface is one of a set of interfaces of an ECMP, then not all packets will be forwarded to this interface, but they will be load-balanced over all possible interfaces. In such case the Multipath Type indicates *how* the packets that will be forwarded to this interface are identified (IP addresses, IP address ranges, bit-masked IP addresses, or bit-masked labels). The actual identification information is then provided in the Multipath Information field.
- Depth Limit: only applicable to a label stack, specifies the maximum number of labels considered in the hash. Should be zero if unspecified or unlimited.
- Multipath Length: the length of the Multipath Information field
- Multipath Information: Address or label values encoded according to the Multipath Type.
- Downstream Label(s): the label(s) in the label stack as it (they) would appear if the packet would be forwarded through this interface. The format is of a MPLS label minus the TTL field.
- Protocol: specifies the protocol that has installed the forwarding entry.

0 for unknown.

The presence of a DSMAP TLV in the MPLS Echo Request, indicates to the replying node that it should include the downstream mapping objects in its MPLS Echo Reply, except if the replying node is the LSP tail-end node for the specified FEC in which case it should not include a DSMAP TLV in its reply.

The usage of the DSMAP TLV is explained with an example. The network topology for this example is the same simple chain of four nodes that was used before, see e.g. Figure 11-2. Node1 in the topology launches an MPLS traceroute for 1.1.1.4/32. The output of the traceroute is displayed in Example 11-9. The `verbose` variant displays a little more information. The character in the first column is the result code, as is explained on top of the command output. The second column is the MPLS TTL of the packet. The third column is the IP address of the node sending the MPLS Echo Reply. The verbose output also shows another IP address: the IP address of the downstream neighbor. The MPLS MTU of the outgoing interface is indicated in the next column (although it says `MRU` in the output), followed by the label stack that would be found on the outgoing packet. If the node is the penultimate hop and PHP is requested then the outgoing label to the downstream neighbor is indicated as `implicit-null`. `Exp:` indicates the value of the EXP bits or TC bits for each label in the stack. Finally the latency is displayed and the numeric return code value if the `verbose` keyword was specified for the command. The downstream neighbor address (fourth column in the output), MTU, and labels are retrieved from the DSMAP TLV in the MPLS Echo Reply packet.

Example 11-9: Example console output of an MPLS traceroute

```
RP/0/0/CPU0:xrvr-1#traceroute mpls ipv4 1.1.1.4/32 fec generic
verbose

Tracing MPLS Label Switched Path to 1.1.1.4/32, timeout is 5
seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output
interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC
mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx
label,
  'P' - no rx intf label prot, 'p' - premature termination of
LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

  0 99.1.2.1 99.1.2.2 MRU 1500 [Labels: 16004 Exp: 0]
L 1 99.1.2.2 99.2.3.3 MRU 1500 [Labels: 16004 Exp: 0] 40 ms,
ret code 8
L 2 99.2.3.3 99.3.4.4 MRU 1500 [Labels: implicit-null Exp: 0]
30 ms, ret code 8
! 3 99.3.4.4 1 ms, ret code 3
```

When a traceroute is executed, this DSMAP TLV is included in the MPLS Echo Request packet such that the intermediate node on the LSP can validate if it is the intended node. Any node that responds to this MPLS Echo Request, except the ultimate node, adds one or more of these TLVs to the MPLS Echo Reply. If the DSMAP TLV in the MPLS Echo Request has Multipath Type set to zero, then the responding node only includes one DSMAP TLV in the Echo Reply, the downstream mapping that would be used by the Echo Request packet. The originating node sets Multipath Type = 0 for a regular traceroute. If the Multipath Type field in the TLV is set to some other value, then the responding node includes all downstream mappings in the Echo Reply. One Downstream Mapping TLV is then added for each outgoing path of the FEC. The originating node sets Multipath Type to a non-zero value for a multipath traceroute, also called "treetrace".

The output of the `mpls traceroute` command displays some of the fields of the Downstream Mapping TLV that were received from the responding nodes: MTU and label stack. The `verbose` output shows even more information of that TLV: IP address of the downstream neighbor.

Example 11-10 shows a packet capture of the MPLS Echo Request originated by Node1 for the tracroute command output in Example 11-9. The packet is captured on the link between Node1 and Node2 in the topology of Figure 11-3. A difference between this packet and an MPLS Echo Request packet used for MPLS ping is the MPLS TTL in line 1, which makes a specific node on the path to process the packet: the node where the TTL expires. Another difference with the packet used for MPLS ping is that the originating node adds a DSMAP TLV to the MPLS Echo Request; see lines 17 to 24. The presence of this TLV will make the responding node to include a DSMAP TLV in its Echo Reply packet. The originating node indicates the intended transit node (the first node on the path in this case since TTL=1) in the DSMAP TLV. In that TLV it indicates that the node where the TTL will expire has:

- Ingress interface with an IPv4 address and an MTU 1500 (line 18)
- IP address 99.1.2.2 (line 19)
- Ingress interface IP address 99.1.2.2 (line 20)
- Label stack of incoming packet is a single label 16004 (line 24)

Example 11-10: MPLS Echo Request packet, TTL 1

```
 1 17:03:31.474163 MPLS (label 16004, exp 0, [S], ttl 1)
 2         IP (tos 0x0, ttl 1, id 169, offset 0, flags [DF], proto UDP (17), length 120, options (RA))
 3     99.1.2.1.3503 > 127.0.0.1.3503:
 4         LSP-PINGv1, msg-type: MPLS Echo Request (1), length: 88
 5           reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
 6           Return Code: No return code or return code contained in the Error Code TLV (0)
 7           Return Subcode: (0)
 8           Sender Handle: 0x000061b2, Sequence: 1
 9           Sender Timestamp: 17:03:04.311512 Receiver Timestamp: no timestamp
10           Vendor Private Code TLV (64512), length: 12
11             Vendor Id: ciscoSystems (9)
12             Value: 0001000400000004
13           Target FEC Stack TLV (1), length: 12
14             Generic IPv4 prefix subTLV (14), length: 5
15               IPv4 Prefix: 1.1.1.4 (1.1.1.4)
16               Prefix Length: 32
17           Downstream Mapping TLV (2), length: 20
18             MTU: 1500, Address-Type: IPv4 Numbered (1)
19             Downstream IP: 99.1.2.2
20             Downstream Interface IP: 99.1.2.2
21             Multipath Type: no multipath (0)
22             Depth Limit: 0
23             Multipath Length: 0
24             Downstream Label Element, Label: 16004, Exp: 0, EOS: 1, Protocol: 0 (Unknown)
```

Node2 verifies if it has an LSP for the target FEC, and if it is the intended transit node. Node2 verifies the elements of the Downstream Mapping TLV and if these elements match with its local information it replies with the applicable return code. In this case Node2 replies with return code 8 (line 5); this means that it switches the packets in the LSP based on the *n*th label in the label stack, with *n* specified in the return sub-code, 1 in this case (line 6). Node2 also includes a Downstream Mapping TLV in the

Echo Reply packet where it provides the downstream information for the LSP. Since Node1 did not specify a Multipath Type (`Multipath Type: no multipath (0)` in line 16) in the request, Node2 only includes a single DSMAP TLV in the reply. The behavior of MPLS traceroute when there is ECMP on the path is discussed in section 11.2.4, “Path Discovery using MPLS Traceroute”. Node2 indicates that the downstream node of the LSP has IP address 99.2.3.3 (line 14); the interface of the downstream node has an IP address 99.2.3.3 (line 15) and the outgoing interface has an MTU 1500 (line 13). The outgoing label stack consists of a single label 16004 (line 19).

Example 11-11: MPLS Echo Reply packet, TTL 1

```
 1| 17:03:33.426948 IP (tos 0xc0, ttl 255, id 24, offset 0,
flags [none], proto UDP (17), length 100)
 2|     99.1.2.2.3503 > 99.1.2.1.3503:
 3|         LSP-PINGv1, msg-type: MPLS Echo Reply (2), length:
72
 4|           reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
 5|           Return Code: Label switched at stack-depth 1 (8)
 6|           Return Subcode: (1)
 7|           Sender Handle: 0x000061b2, Sequence: 1
 8|           Sender Timestamp: 17:03:04.311512 Receiver
Timestamp: 17:03:04.327131
 9|           Vendor Private Code TLV (64512), length: 12
10|             Vendor Id: ciscoSystems (9)
11|             Value: 0001000400000004
12|           Downstream Mapping TLV (2), length: 20
13|             MTU: 1500, Address-Type: IPv4 Numbered (1)
14|             Downstream IP: 99.2.3.3
15|             Downstream Interface IP: 99.2.3.3
16|             Multipath Type: no multipath (0)
17|             Depth Limit: 0
18|             Multipath Length: 0
19|             Downstream Label Element, Label: 16004, Exp: 0,
EOS: 1, Protocol: 0 (Unknown)
```

Node1 receives the downstream mapping information from Node2 and includes that information in the next MPLS Echo Request packet, targeted at the next node of the LSP, which is Node3. See Example 11-12. Node1 sets TTL=2 in that packet (line 1), using the same label 16004.

Example 11-12: MPLS Echo Request packet, TTL 2

```
 1| 17:03:33.440386 MPLS (label 16004, exp 0, [S], ttl 2)
 2|         IP (tos 0x0, ttl 1, id 170, offset 0, flags [DF],
proto UDP (17), length 120, options (RA))
 3|     99.1.2.1.3503 > 127.0.0.1.3503:
 4|         LSP-PINGv1, msg-type: MPLS Echo Request (1), length:
88
 5|           reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
 6|           Return Code: No return code or return code
contained in the Error Code TLV (0)
 7|           Return Subcode: (0)
 8|           Sender Handle: 0x000061b2, Sequence: 2
 9|           Sender Timestamp: 17:03:06.294275 Receiver
Timestamp: no timestamp
10|           Vendor Private Code TLV (64512), length: 12
11|             Vendor Id: ciscoSystems (9)
12|             Value: 0001000400000004
13|           Target FEC Stack TLV (1), length: 12
14|             Generic IPv4 prefix subTLV (14), length: 5
15|               IPv4 Prefix: 1.1.1.4 (1.1.1.4)
16|               Prefix Length: 32
17|           Downstream Mapping TLV (2), length: 20
18|             MTU: 1500, Address-Type: IPv4 Numbered (1)
19|             Downstream IP: 99.2.3.3
20|             Downstream Interface IP: 99.2.3.3
21|             Multipath Type: no multipath (0)
22|             Depth Limit: 0
23|             Multipath Length: 0
24|             Downstream Label Element, Label: 16004, Exp: 0,
EOS: 1, Protocol: 0 (Unknown)
```

The Echo Request packet’s TTL expires on Node3. Node3 verifies the FEC and if it is the intended node, and responds with an MPLS Echo Reply. See Example 11-13. Node3 specifies the downstream information in the DSMAP TLV. Since Node3 is the penultimate hop for this LSP (Prefix-SID of Node4), the outgoing label stack is `Implicit-Null` (line 19). Node3 pops the label before forwarding the packet to Node4.

Example 11-13: MPLS Echo Reply packet, TTL 2

```
 1| 17:03:35.346122 IP (tos 0xc0, ttl 254, id 32, offset 0,
flags [none], proto UDP (17), length 100)
 2|     99.2.3.3.3503 > 99.1.2.1.3503:
 3|         LSP-PINGv1, msg-type: MPLS Echo Reply (2), length:
72
 4|           reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
 5|           Return Code: Label switched at stack-depth 1 (8)
 6|           Return Subcode: (1)
 7|           Sender Handle: 0x000061b2, Sequence: 2
 8|           Sender Timestamp: 17:03:06.294275 Receiver
Timestamp: 17:03:06.405017
 9|           Vendor Private Code TLV (64512), length: 12
10|             Vendor Id: ciscoSystems (9)
11|             Value: 0001000400000004
12|           Downstream Mapping TLV (2), length: 20
13|             MTU: 1500, Address-Type: IPv4 Numbered (1)
14|             Downstream IP: 99.3.4.4
15|             Downstream Interface IP: 99.3.4.4
16|             Multipath Type: no multipath (0)
17|             Depth Limit: 0
18|             Multipath Length: 0
19|             Downstream Label Element, Label: 3 (Implicit-
Null), Exp: 0, EOS: 1, Protocol: 0 (Unknown)
```

Node1 receives the downstream mapping information from Node3 and includes that information in the next MPLS Echo Request packet, targeted at the next node of the LSP, which is Node4. See Example 11-14. Node1 sets TTL=3 in that packet (line 1).

Example 11-14: MPLS Echo Request packet, TTL 3

```
 1| 17:03:35.352786 MPLS (label 16004, exp 0, [S], ttl 3)
 2|         IP (tos 0x0, ttl 1, id 171, offset 0, flags [DF],
proto UDP (17), length 120, options (RA))
```

```
 3      99.1.2.1.3503 > 127.0.0.1.3503:
 4          LSP-PINGv1, msg-type: MPLS Echo Request (1), length:
88
 5            reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
 6            Return Code: No return code or return code
contained in the Error Code TLV (0)
 7            Return Subcode: (0)
 8            Sender Handle: 0x000061b2, Sequence: 3
 9            Sender Timestamp: 17:03:08.259859 Receiver
Timestamp: no timestamp
10            Vendor Private Code TLV (64512), length: 12
11              Vendor Id: ciscoSystems (9)
12              Value: 0001000400000004
13            Target FEC Stack TLV (1), length: 12
14              Generic IPv4 prefix subTLV (14), length: 5
15                IPv4 Prefix: 1.1.1.4 (1.1.1.4)
16                Prefix Length: 32
17            Downstream Mapping TLV (2), length: 20
18              MTU: 1500, Address-Type: IPv4 Numbered (1)
19              Downstream IP: 99.3.4.4
20              Downstream Interface IP: 99.3.4.4
21              Multipath Type: no multipath (0)
22              Depth Limit: 0
23              Multipath Length: 0
24              Downstream Label Element, Label: 3 (Implicit-
Null), Exp: 0, EOS: 1, Protocol: 0 (Unknown)
```

Node4 verifies the FEC and the downstream mapping information. Node4 then confirms with the return code 3 that is the tail-end of the LSP (line 5). It is an egress node of the FEC 1.1.1.4/32 at a stack depth indicated in the return sub-code, 1 in this case.

Example 11-15: MPLS Echo Reply packet, TTL 3

```
 1 17:03:35.363333 IP (tos 0xc0, ttl 253, id 27, offset 0,
flags [none], proto UDP (17), length 76)
 2      99.3.4.4.3503 > 99.1.2.1.3503:
 3          LSP-PINGv1, msg-type: MPLS Echo Reply (2), length:
48
 4            reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
 5            Return Code: Replying router is an egress for the
```

```
FEC at stack depth 1 (3)
            Return Subcode: (1)
            Sender Handle: 0x000061b2, Sequence: 3
            Sender Timestamp: 17:03:08.259859 Receiver
Timestamp: 17:03:08.272629
            Vendor Private Code TLV (64512), length: 12
              Vendor Id: ciscoSystems (9)
              Value: 0001000400000004
```

HIGHLIGHT: MPLS ping and traceroute

Provide enhanced connectivity checks, error isolation and path verification abilities specifically for MPLS LSPs including Segment Routing

Verification includes both control plane and data plane verification and consistency checks

Include path discovery mechanisms that allow verification of ECMP paths in the network

Segment routing LSPs can be verified using Generic FEC and Nil FEC while the new extensions to LSP ping protocol are being standardized.

11.2.4 Path Discovery using MPLS Traceroute

The ability to discover and trace all possible paths a packet may traverse to get to its final destination is an important characteristic required for troubleshooting LSP failures. The transit node's ECMPs are not visible to the original sender of a packet, so it is important to be able to discover and validate all the possible paths through the network from any given source node.

MPLS traceroute is designed to discover all ECMPs on the path to the destination. A node load-balances the traffic over all available equal cost paths based on the header fields of the packet: label stack, source and destination IP addresses, layer-4 ports, etc. MPLS traceroute can send the MPLS Echo Request packets with any destination IP address from the address range 127.0.0.0/8. Therefore, the originating node of the MPLS Echo Request packets can send the packets on the different equal cost paths by varying the IP destination address of the packets.

To discover the paths, the source node sends a bitmap in the Multipath Info sub-TLV of the DSMAP TLV. Together with a base address, this bitmap indicates which IP destination addresses are hashed on the path identified by the downstream mapping information.

Each bit in the bitmap represents an address that is hashed on the path. The address is the base address plus the position of the bit in the mask, starting with zero. For example, base address 127.0.0.1 and bitmask 10001001011001001010100100010101 indicates that the following addresses from the range 127.0.0.1-127.0.0.32 are hashed on this path: 127.0.0.1, 127.0.0.5, 127.0.0.8, 127.0.0.10, 127.0.0.11, 127.0.0.14, 127.0.0.17, 127.0.0.19, 127.0.0.21, 127.0.0.24, 127.0.0.28, 127.0.0.30, and 127.0.0.32.

The source node starts the discovery process by sending a bitmap in Multipath Info sub-TLV in the DSMAP TLV. The initial bitmap (0xFFFFFFFF) contains all addresses. The transit node queries the forwarding table for each address in the bitmap to identify the outgoing path (the downstream mapping). For each address it sets the corresponding bit in the bitmap of the corresponding DSMAP TLV in the MPLS Echo Reply packet and returns the packet to the source node. The source node uses this information to interrogate each successive router along the path, branching the bitmap into further sub-sets. The process is repeated until the destination is reached for each path.

The multipath discovery process is illustrated using the topology in Figure 11-6. Before looking at the actual multipath discovery, retrieval of the DSMAP information for FEC 1.1.1.4/32 of Node2 is illustrated by using MPLS ping.

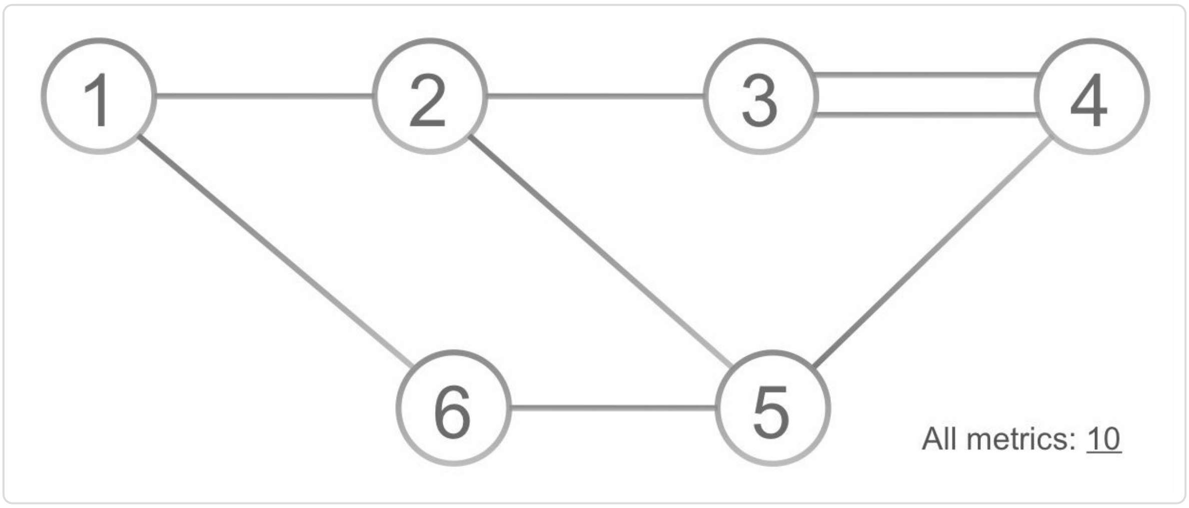


Figure 11-6: Multipath discovery example topology

The user executes an MPLS ping (see Example 11-16) and specifies she wants to see the DSMAP information (`dsmap`) for FEC 1.1.1.4/32 on the first node (`ttl 1`) on the LSP going via Node2 (`output nexthop 99.1.2.2`). The TTL of the packet expires on Node2 and Node2 sends an MPLS Echo Reply containing the DSMAP TLVs of all its paths to 1.1.1.4/32: via Node3 and via Node5. The DSMAP information of the first path is shown in lines 17 to 23 and the DSMAP information of the second path is shown in lines 24 to 31. The output of the MPLS ping command displays the destination IP addresses it computed from the bitmap of the DSMAP.

Example 11-16: DSMAP retrieval using MPLS ping

```
 1| RP/0/0/CPU0:xrvr-1#ping mpls ipv4 1.1.1.4/32 fec generic
dsmap ttl 1 output nexthop 99.1.2.2 repeat 1
 2|
Sending 1, 100-byte MPLS Echos to 1.1.1.4/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

L Echo Reply received from 99.1.2.2
  DSMAP 0, DS Router Addr 99.2.3.3, DS Intf Addr 99.2.3.3
    Depth Limit 0, MRU 1500 [Labels: 16004 Exp: 0]
    Multipath Addresses:
      127.0.0.1             127.0.0.5             127.0.0.8             127.0.0.10
      127.0.0.11            127.0.0.14            127.0.0.17            127.0.0.19
      127.0.0.21            127.0.0.24            127.0.0.28            127.0.0.30
      127.0.0.32
  DSMAP 1, DS Router Addr 99.2.5.5, DS Intf Addr 99.2.5.5
    Depth Limit 0, MRU 1500 [Labels: 16004 Exp: 0]
    Multipath Addresses:
      127.0.0.2             127.0.0.3             127.0.0.4             127.0.0.6
      127.0.0.7             127.0.0.9             127.0.0.12            127.0.0.13
      127.0.0.15            127.0.0.16            127.0.0.18            127.0.0.20
      127.0.0.22            127.0.0.23            127.0.0.25            127.0.0.26
      127.0.0.27            127.0.0.29            127.0.0.31

Success rate is 0 percent (0/1)
```

The packet capture of the MPLS Echo Request packet is shown in Example 11-17. Specific to this packet is the Downstream Mapping TLV in lines 17 to 26. Since Node1 does not include a label stack in the DSMAP TLV and sets the downstream node's address to 224.0.0.2 and the interface index to 0. The Multipath Info bitmap includes all addresses.

Example 11-17: Captured MPLS Echo Request packet requesting DSMAP

```
 1 14:21:26.780585 MPLS (label 16004, exp 0, [S], ttl 1)
 2         IP (tos 0x0, ttl 1, id 41, offset 0, flags [DF],
proto UDP (17), length 124, options (RA))
 3     99.1.2.1.3503 > 127.0.0.1.3503:
 4         LSP-PINGv1, msg-type: MPLS Echo Request (1), length:
92
 5           reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
 6           Return Code: No return code or return code
contained in the Error Code TLV (0)
 7           Return Subcode: (0)
 8           Sender Handle: 0x00004f76, Sequence: 1
 9           Sender Timestamp: 14:21:07.148877 Receiver
Timestamp: no timestamp
10           Vendor Private Code TLV (64512), length: 12
11             Vendor Id: ciscoSystems (9)
12             Value: 0001000400000004
13           Target FEC Stack TLV (1), length: 12
14             Generic IPv4 prefix subTLV (14), length: 5
15               IPv4 Prefix: 1.1.1.4 (1.1.1.4)
16               Prefix Length: 32
17           Downstream Mapping TLV (2), length: 24
18             MTU: 0, Address-Type: IPv4 Unnumbered (2)
19             Downstream IP: 224.0.0.2
20             Upstream Interface Index: 0x00000000
21             Multipath Type: Bit-masked IPv4 address set (8)
22             Depth Limit: 0
23             Multipath Length: 8
24             Multipath Information
25                 IP Address: 127.0.0.1 (127.0.0.1)
26                 Mask: ffffffff
```

Node2 replies with all possible paths included as DSMAP TLVs in the MPLS Echo Reply packet. This packet is shown in Example 11-18. The first DSMAP TLV is displayed in lines 12 to 22 and the second in lines 24 to 34.

Example 11-18: Captured MPLS Echo Reply packet multipath DSMAP

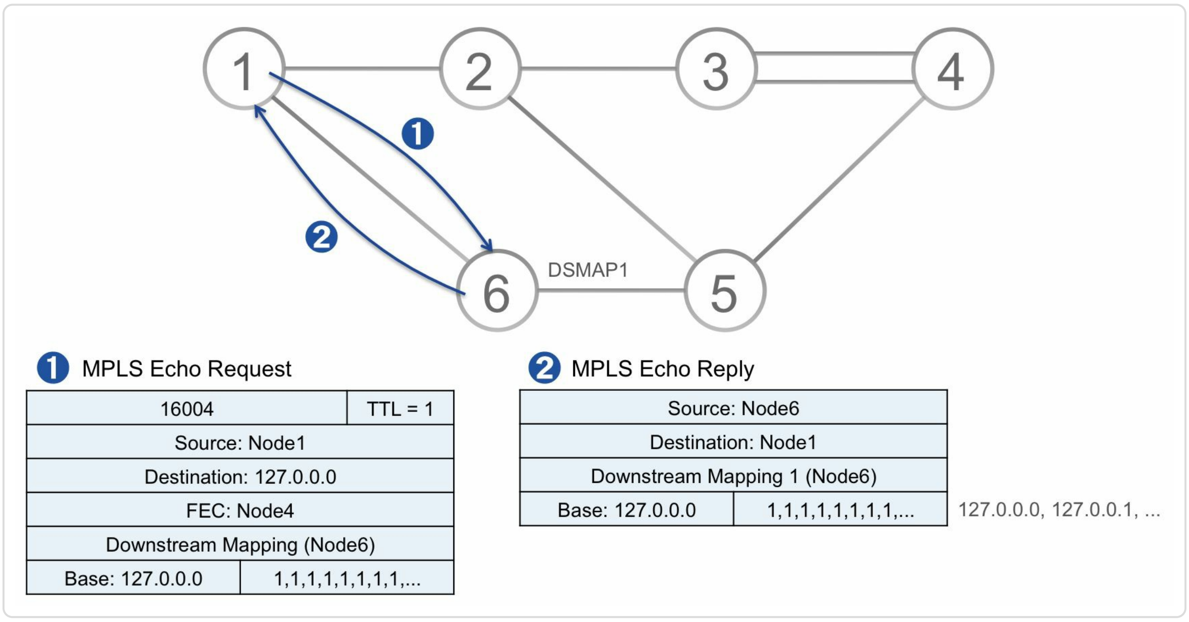
```
 1| 14:21:26.782376 IP (tos 0xc0, ttl 255, id 19, offset 0,
flags [none], proto UDP (17), length 140)
 2|     99.1.2.2.3503 > 99.1.2.1.3503:
 3|         LSP-PINGv1, msg-type: MPLS Echo Reply (2), length:
112
 4|           reply-mode: Reply via an IPv4/IPv6 UDP packet (2)
 5|           Return Code: Label switched at stack-depth 1 (8)
 6|           Return Subcode: (1)
 7|           Sender Handle: 0x00004f76, Sequence: 1
 8|           Sender Timestamp: 14:21:07.148877 Receiver
Timestamp: 14:21:05.127492
 9|           Vendor Private Code TLV (64512), length: 12
10|             Vendor Id: ciscoSystems (9)
11|             Value: 0001000400000004
12|           Downstream Mapping TLV (2), length: 28
13|             MTU: 1500, Address-Type: IPv4 Numbered (1)
14|             Downstream IP: 99.2.3.3
15|             Downstream Interface IP: 99.2.3.3
16|             Multipath Type: Bit-masked IPv4 address set (8)
17|             Depth Limit: 0
18|             Multipath Length: 8
19|             Multipath Information
20|                 IP Address: 127.0.0.1 (127.0.0.1)
21|                 Mask: 8964a915
22|             Downstream Label Element, Label: 16004, Exp: 0,
EOS: 1, Protocol: 0 (Unknown)
23|
24|           Downstream Mapping TLV (2), length: 28
25|             MTU: 1500, Address-Type: IPv4 Numbered (1)
26|             Downstream IP: 99.2.5.5
27|             Downstream Interface IP: 99.2.5.5
28|             Multipath Type: Bit-masked IPv4 address set (8)
29|             Depth Limit: 0
30|             Multipath Length: 8
31|             Multipath Information
```

```
                    IP Address: 127.0.0.1 (127.0.0.1)
                    Mask: 769b56ea
               Downstream Label Element, Label: 16004, Exp: 0,
EOS: 1, Protocol: 0 (Unknown)
```

In the multipath discovery example, the user starts a multipath traceroute on Node1 for the FEC bound to Node4's loopback prefix 1.1.1.4/32.

Node1 itself has two equal cost paths to 1.1.1.4/32: via Node2 and via Node6. Traceroute first explores the path to 1.1.1.4/32 via Node6, using IP destination address 127.0.0.0.

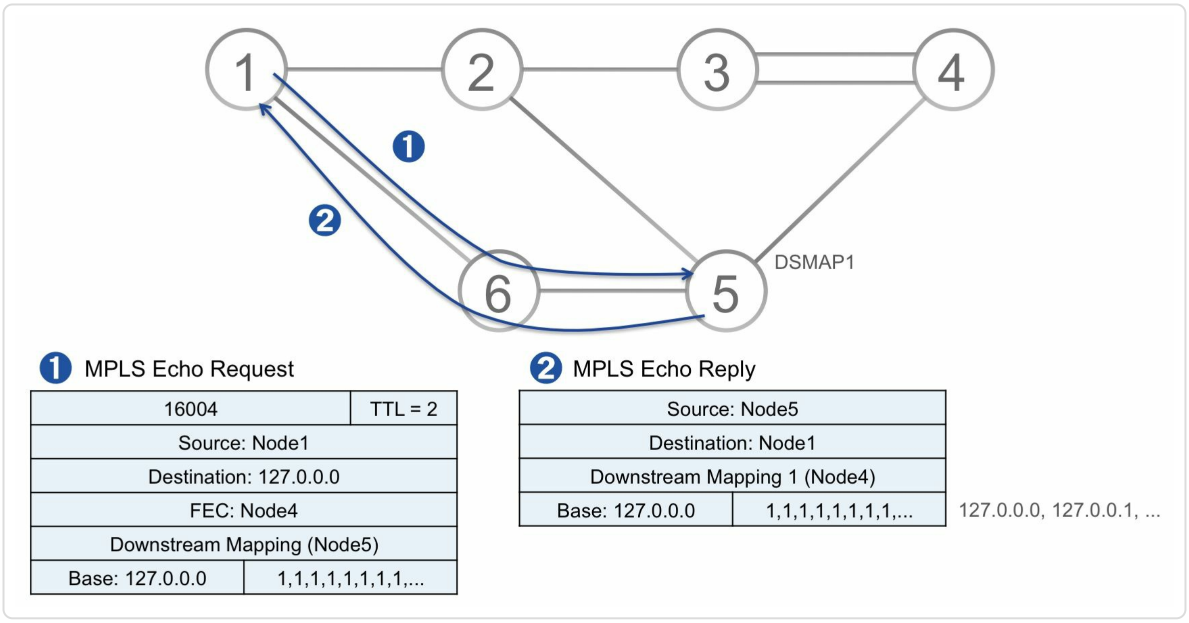
Node1 sends an MPLS Echo Request with destination address 127.0.0.0 and TTL=1 to Node6. The TTL expires on Node6 and Node6 returns an Echo Reply containing a single DSMAP TLV, for the path via Node5. The address bitmap in that DSMAP TLV contains all addresses since there is only a single path. See Figure 11-7.



16004	TTL = 1
Source: Node1	
Destination: 127.0.0.0	
FEC: Node4	
Downstream Mapping (Node6)	
Base: 127.0.0.0	1,1,1,1,1,1,1,1,...

Source: Node6	
Destination: Node1	
Downstream Mapping 1 (Node6)	
Base: 127.0.0.0	1,1,1,1,1,1,1,1,...

Figure 11-7: Multipath traceroute illustration – step 1



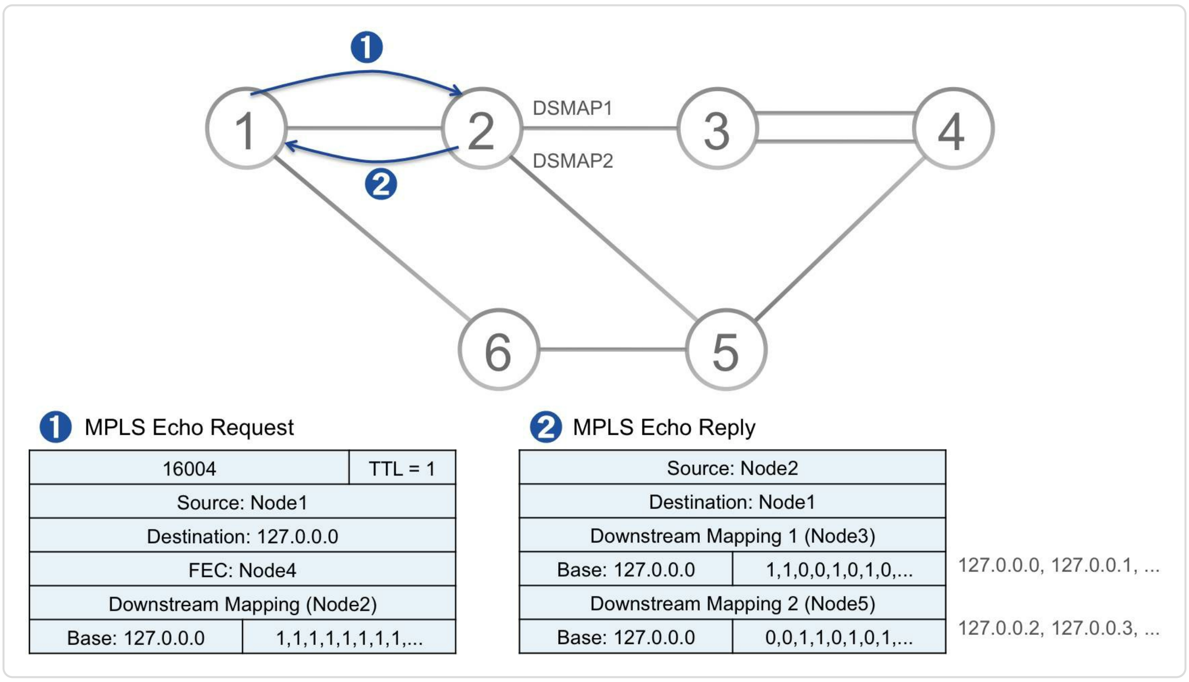
❶ MPLS Echo Request

16004	TTL = 2
Source: Node1	
Destination: 127.0.0.0	
FEC: Node4	
Downstream Mapping (Node5)	
Base: 127.0.0.0	1,1,1,1,1,1,1,1,...

❷ MPLS Echo Reply

Source: Node5	
Destination: Node1	
Downstream Mapping 1 (Node4)	
Base: 127.0.0.0	1,1,1,1,1,1,1,1,...

Figure 11-8: Multipath traceroute illustration – step 2



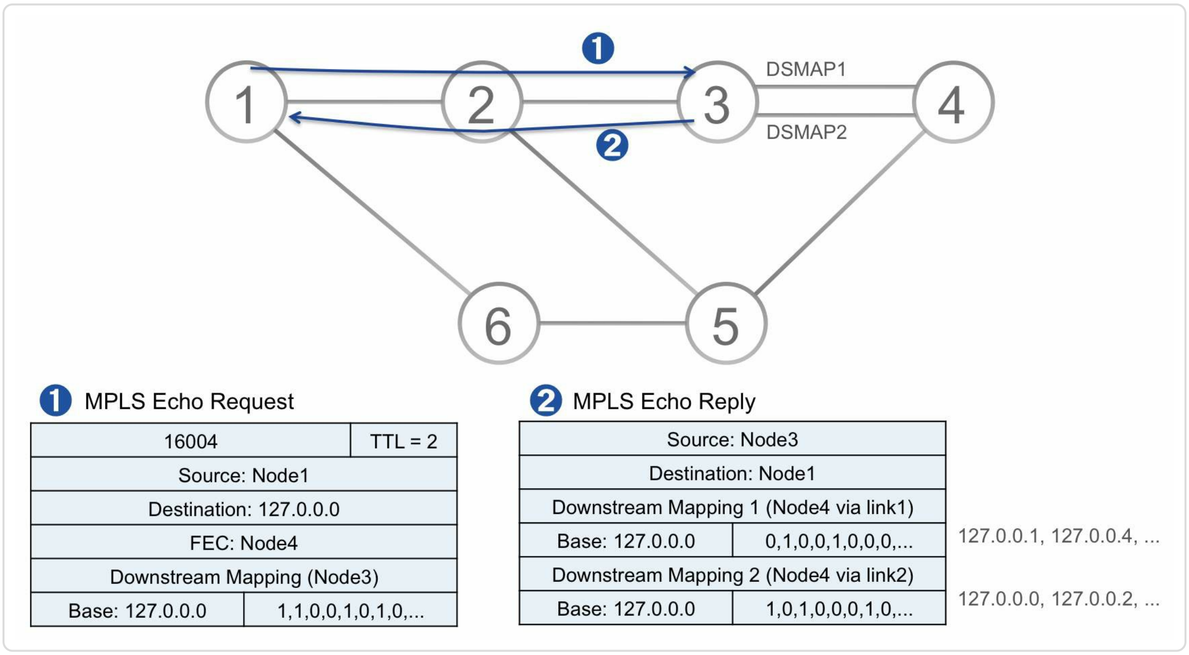
❶ MPLS Echo Request

16004	TTL = 1
Source: Node1	
Destination: 127.0.0.0	
FEC: Node4	
Downstream Mapping (Node2)	
Base: 127.0.0.0	1,1,1,1,1,1,1,1,...

❷ MPLS Echo Reply

Source: Node2	
Destination: Node1	
Downstream Mapping 1 (Node3)	
Base: 127.0.0.0	1,1,0,0,1,0,1,0,...
Downstream Mapping 2 (Node5)	
Base: 127.0.0.0	0,0,1,1,0,1,0,1,...

Figure 11-9: Multipath traceroute illustration – step 3



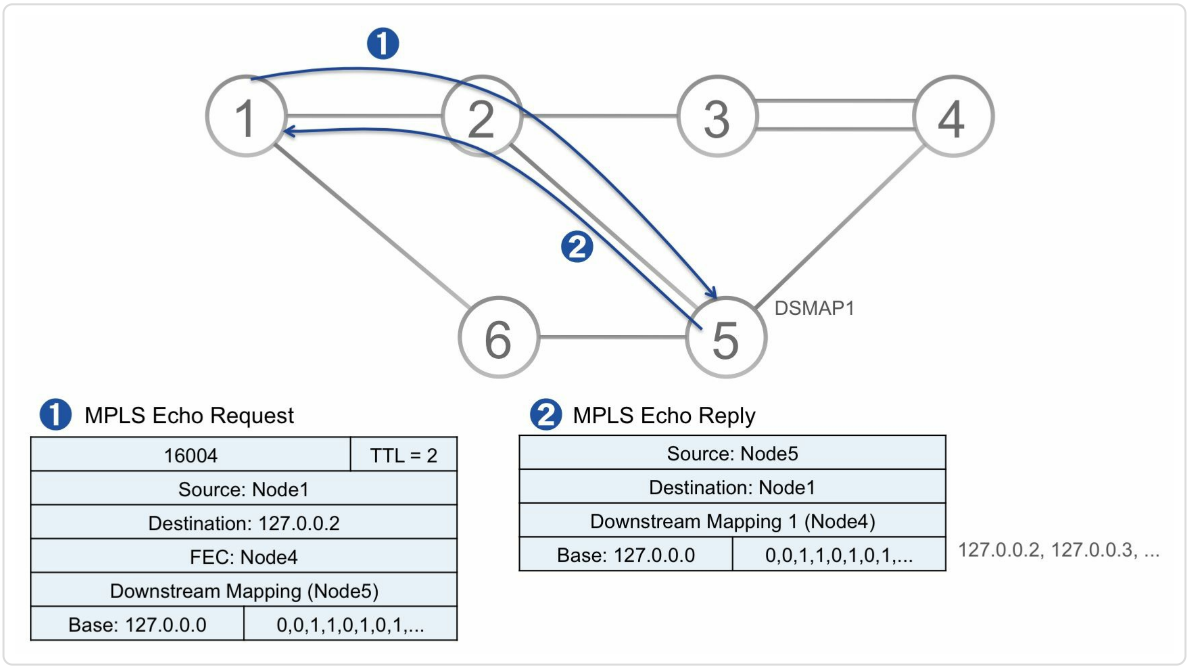
1 MPLS Echo Request

16004	TTL = 2
Source: Node1	
Destination: 127.0.0.0	
FEC: Node4	
Downstream Mapping (Node3)	
Base: 127.0.0.0	1,1,0,0,1,0,1,0,...

2 MPLS Echo Reply

Source: Node3		
Destination: Node1		
Downstream Mapping 1 (Node4 via link1)		
Base: 127.0.0.0	0,1,0,0,1,0,0,0,...	127.0.0.1, 127.0.0.4, ...
Downstream Mapping 2 (Node4 via link2)		
Base: 127.0.0.0	1,0,1,0,0,0,1,0,...	127.0.0.0, 127.0.0.2, ...

Figure 11-10: Multipath traceroute illustration – step 4



1 MPLS Echo Request

16004	TTL = 2
Source: Node1	
Destination: 127.0.0.2	
FEC: Node4	
Downstream Mapping (Node5)	
Base: 127.0.0.0	0,0,1,1,0,1,0,1,...

2 MPLS Echo Reply

Source: Node5		
Destination: Node1		
Downstream Mapping 1 (Node4)		
Base: 127.0.0.0	0,0,1,1,0,1,0,1,...	127.0.0.2, 127.0.0.3, ...

Figure 11-11: Multipath traceroute illustration – step 5

- Node1→Node6→Node5→Node4: 127.0.0.0 via Gi0/0/0/0
- Node1→Node2→Node3→Node4(link1): 127.0.0.1 via Gi0/0/0/1
- Node1→Node2→Node3→Node4(link2): 127.0.0.0 via Gi0/0/0/1

- Node1→Node2→Node5→Node4: 127.0.0.2 via Gi0/0/0/1

Example 11-19: MPLS multipath traceroute console output example

```
RP/0/0/CPU0:xrvr-1#traceroute mpls multipath ipv4 1.1.1.4/32
fec generic

Starting LSP Path Discovery for 1.1.1.4/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output
interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC
mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx
label,
  'P' - no rx intf label prot, 'p' - premature termination of
LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

LL!
Path 0 found,
 output interface GigabitEthernet0/0/0/0 nexthop 99.1.6.6
 source 99.1.6.1 destination 127.0.0.0
LL!
Path 1 found,
 output interface GigabitEthernet0/0/0/1 nexthop 99.1.2.2
 source 99.1.2.1 destination 127.0.0.1
!
Path 2 found,
 output interface GigabitEthernet0/0/0/1 nexthop 99.1.2.2
 source 99.1.2.1 destination 127.0.0.0
L!
Path 3 found,
 output interface GigabitEthernet0/0/0/1 nexthop 99.1.2.2
 source 99.1.2.1 destination 127.0.0.2

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (9/0)
 Echo Reply (received/timeout) (9/0)
 Total Time Elapsed 89 ms
```

The user can now exercise any of the paths from Node1 to 1.1.1.4/32 by using the destination addresses and nexthop information provided by the multipath traceroute. For example, to send a packet to 1.1.1.4/32 via the path Node1→Node2→Node5→Node4 the user can specify destination address 127.0.0.2 and outgoing interface Gi0/0/0/1 and/or nexthop 99.1.2.2 (Node2). This is illustrated in Example 11-20. Remember that in the IP addressing convention, the last digit of an interface IP address is the number of the node.

Example 11-20: MPLS traceroute along specific path

```
RP/0/0/CPU0:xrvr-1#traceroute mpls ipv4 1.1.1.4/32 fec generic
destination 127.0.0.2 output nexthop 99.1.2.2

Tracing MPLS Label Switched Path to 1.1.1.4/32, timeout is 2
seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output
interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC
mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx
label,
  'P' - no rx intf label prot, 'p' - premature termination of
LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

  0 99.1.2.1 MRU 1500 [Labels: 16004 Exp: 0]
L 1 99.1.2.2 MRU 1500 [Labels: 16004 Exp: 0] 10 ms
L 2 99.2.5.5 MRU 1500 [Labels: implicit-null Exp: 0] 10 ms
! 3 99.4.5.4 10 ms
```

11.2.5 MPLS Ping and Traceroute using Nil-FEC

Since MPLS Ping and Traceroute interact with their corresponding control plane for path verification of an LSP, supporting a new technology such as Segment Routing may require changes or additions to these interactions or to the tools themselves.

Operators require a basic data plane verification tool before the above changes to the tools have been standardized and developed. This is required since IP ping and traceroute are not sufficient in an MPLS network since they may not detect MPLS fault, as we’ve seen in the beginning of this chapter.

A FEC named “Nil-FEC” is defined in IETF RFC 4379. Using this FEC specifies that no explicit FEC (Control plane) is associated with the label of the packet. One can therefore use existing ping and traceroute infrastructure with a Nil-FEC in the Target FAC Stack TLV.

Nil-FEC ping and traceroute provides a basic mechanism to verify the data plane by specifying a label stack on the command line to reach the destination. The MPLS Echo Request packet gets label switched to the destination using MPLS labels only. Using Nil-FEC, there is no interaction with the control plane, so no software changes are required. It supports new and emerging technologies (including Segment Routing) as the path is only specified as a label stack.

11.3 MPLS OAM for Segment Routing

The Segment Routing extensions for the LSP ping protocol are under development and being specified at the IETF as part of draft-ietf-mpls-spring-lsp-ping. At the time of writing this book, the specification work was still in progress and the implementation not yet released on Cisco IOS XR platforms. We will cover these in more detail as the specifications and implementations mature in perhaps a future revision of this book.

In the meantime, SR MPLS paths can be verified by using the existing MPLS OAM tools using Nil-FEC or generic FEC, as described earlier in this chapter.

11.4 Summary

- The well-known IP ping and traceroute tools can be used in an MPLS network, but are not sufficient to diagnose all MPLS specific problems.
- IP ping and traceroute rely on ICMP functionality. ICMP has been extended to provide more MPLS related information for IP traceroute.
- MPLS ping and traceroute are specifically designed to diagnose MPLS transport problems and consistency between control plane and data plane.
- The MPLS tools use MPLS Echo Request and Reply messages carried in UDP packets in an extensible TLV format that allows for enhanced capabilities for path discovery and verification.
- While SR MPLS specific extensions for the MPLS tools are under development, Generic FEC and Nil-FEC can be used to diagnose failures in an SR MPLS network.

11.5 References

- [draft-ietf-mpls-spring-lsp-ping], Kumar, N., Swallow, G., Pignataro, C., Akiya, N., Kini, S., Gredler, H., and M. Chen, "Label Switched Path (LSP) Ping/Trace for Segment Routing Networks Using MPLS Dataplane", draft-ietf-mpls-spring-lsp-ping (work in progress), May 2016, https://datatracker.ietf.org/doc/draft-ietf-mpls-spring-lsp-ping.
- [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, https://datatracker.ietf.org/doc/rfc792.
- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, https://datatracker.ietf.org/doc/rfc3032.
- [RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures", RFC 4379, DOI 10.17487/RFC4379, February 2006, https://datatracker.ietf.org/doc/rfc4379.
- [RFC4950] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "ICMP Extensions for Multiprotocol Label Switching", RFC 4950, DOI 10.17487/RFC4950, August 2007, https://datatracker.ietf.org/doc/rfc4950.

---

[1] UDP is used for historical reasons. The router vendors had initially implemented a strict interpretation of RFC 792 that says: "(…) no ICMP messages are sent about ICMP messages." Due to this, Van Jacobson wrote a working traceroute version that used UDP probe packets instead of

ICMP. Later, RFC 1122 section 3.2.2 indicated that ICMP messages must not be sent in response to ICMP *error* messages. Also see http://www.inetdaemon.com/tutorials/troubleshooting/tools/traceroute/defi

[2] This is in contrast with IETF RFC 1812 that specifies: "(…) the IP source address in an ICMP message originated by the router MUST be one of the IP addresses associated with the physical interface over which the ICMP message is transmitted."

[3] http://www.iana.org/assignments/mpls-lsp-ping-parameters/mpls-lsp-ping-parameters.xhtml

[4] Since the IPv4 Explicit-Null is added the Nil-FEC is not usable to verify IPv6 signaled MPLS paths.

12 SEGMENT ROUTING IPV6 DATE PLANE

Segment Routing applies to both IPv6 and MPLS data planes. The difference between the two is not in the architecture but in the implementation of that architecture.

When using the MPLS data plane to transport IPv6 packets, the Segment List or SID List is imposed on the packet header in the form of a stack of MPLS labels. Each label represents a segment and the top label represents the Active Segment. When a segment has completed then the label representing that segment is removed from the labels stack. Segment Routing MPLS data plane uses the existing MPLS architecture.

When using the Segment Routing IPv6 data plane to transport IPv6 packets (commonly called SRv6), the Segment List is imposed on the packet header in a Segment Routing Header (SRH). This header is a new type of Routing Header, a type of Extension Header described in the IPv6 specification IETF RFC 2460. A pointer in the SRH points to the Active Segment in the list of segments encoded in the header. When a segment has completed then the segment is not removed from the list, but the pointer is updated to point to the next segment in the list.

To enable SRv6 in a network, only the nodes that have to process the packet header must have SRv6 data plane support. All other nodes in the network can be just plain IPv6 nodes. SRv6 does not require a forklift upgrade of the network since SRv6 and non-SRv6 nodes are fully interoperable, they can co-exist in the network. This makes a gradual deployment of SRv6 in the network possible.

In this chapter, we provide an overview of SRv6 and its fundamental building blocks. As we go through the rest of the book that is focused on MPLS data plane, it would become evident to the discerning reader that all of those concepts, mechanisms and abilities also apply to SRv6 – the Segment Routing architecture is common for both data planes. Implementation details and deployment models would defer. More details on these aspects of SRv6 would be covered in a future part of this book.

12.1 “Segment” in IPv6

The notion of a “segment” is not new in IPv6. The term “segment” is used in the Routing Header section of the IPv6 protocol specification IETF RFC 2460, where it is used to indicate the stretch (or span) of a source routed path between two hops of that path.

In Segment Routing, a segment can be any instruction: topological, service, etc.

Figure 12-1 shows the different types of nodes on a Segment Routing IPv6 path: Node11 is the Source Node; Node2, Node6, and Node12 are Segment Endpoint Nodes; Node12 is also destination node; Node1, Node4, and Node7 are Transit Nodes.

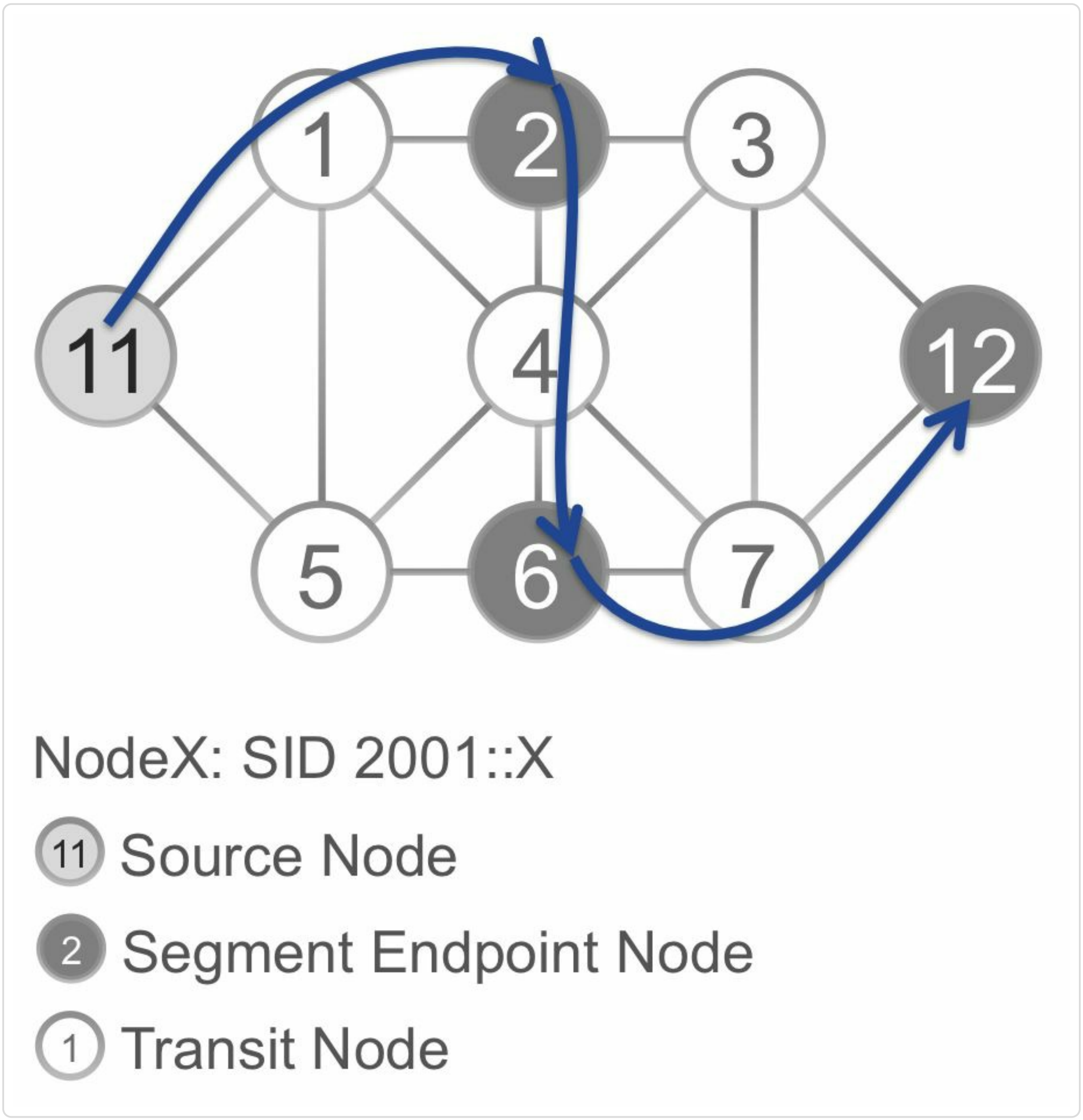


Figure 12-1: SRv6 path

HIGHLIGHT

With SRv6, truly local segments will be rarely used.

For example, an SRv6 Adj-SID is global to its SR domain. The SR domain routes the packets with the SID up to the node that originates it and this node applies the instruction (function) locally associated with the Adj-SID: i.e. activate the next SID and forward the packet along the associated adjacency/interface.

An SRv6 Adj-SID can thus be seen as a Prefix-SID whose function is "forward on this adjacency".

In SRv6, we will use many different forms of functions (instructions) associated with a Prefix-SID: e.g. lookup the next active SID in a specific VRF, duplicate the packet and send each copy on two disjoint paths related to the SID, etc.

Source Node

A Source Node can be any node originating an IPv6 packet with a Segment Routing Header:

- A node originating an IPv6 packet with SRH
- An ingress node (in many general cases a router) of a SRv6 Domain that inserts a SRH in the IPv6 header or imposes an outer IPv6 header with SRH on the packet

Segment Endpoint Node

A Segment Endpoint Node is the node that terminates a segment. The Destination Address of an IPv6 packet with SRH corresponds to an address of the Segment Endpoint Node. The Segment Endpoint Node examines and updates the SRH and applies the instruction associated with the Active Segment, the segment of the Destination Address of the packet.

Transit Node

A Transit Node is a node (more specifically a router) on the path of the packet, and that is not a Segment Endpoint Node. The Destination Address of the packet does not correspond to an address of a Transit Node.

The IPv6 specification (IETF RFC 2460) specifies that a node may not process the Routing Header if that node is not corresponding to the Destination Address of the packet. A Transit Node forwards the IPv6 packet based on its IPv6 Destination Address without looking at the SRH or touching it. Therefore, a Transit Node does not need to support SRv6 to forward the packet.

12.2 Segment Identifiers in SRv6

Segments in SRv6 are identified by 128 bits IPv6 addresses. From a signaling perspective this is much simpler compared to the MPLS data plane: no need to advertise anything else than IPv6 prefixes. The prefix is the Segment Identifier (SID). An IPv6 address may represent more than just a router. It can represent an interface, an appliance, a service, an application, etc. Or it can represent a set of any of these.

IPv6 addresses can be summarized, that also applies to the IPv6 addresses used as SID. Summarization is not possible for MPLS SIDs.

12.3 IPv6 Headers Reminder

IPv6 uses two distinct types of headers: (main) IPv6 Header and IPv6 Extension Headers. The main IPv6 header is equivalent to the basic IPv4 header. Comparing the IPv6 header to the IPv4 header, a number of fields have been changed and a number of fields have been removed.

One of the removed fields is the IPv4 options field. The options field that existed in the IPv4 header was used to convey additional information about the packet, or about the way the packet should be processed. The IP Router Alert Option (IETF RFC 2113) is probably the best known and most used option, it alerts transit routers to examine the contents of the IPv4 packet. In IPv6 the functionality of options is removed from the main header and implemented through a set of additional headers called Extension Headers (IETF RFC 2460). Therefore the main IPv6 header has a fixed size (40 bytes) while customized Extension Headers are added as needed.

The Protocol Type field in the IPv4 header that is used to indicate which upper layer protocol header comes after the IPv4 header, has been renamed to Next Header field in the IPv6 header.

IPv6 Extension Header

The use of Extension Headers allows extending IPv6 to support future needs and capabilities. An IPv6 packet may carry zero, one, or more Extension Headers of varying lengths. In a typical IPv6 packet no Extension Headers are present. If the packet requires special handling from either intermediate routers on its path or the destination node, then one or more Extension Headers are added in the packet header.

The Extension Headers are located between the main IPv6 Header and the upper-layer header in a packet. A small number of Extension Header types exist, each identified by a distinct type value. If an Extension Header is present in the header, then the Next Header field in the IPv6 header contains the type of the Extension header, for example type 43 for the Routing Header. The Next Header field of the Extension header indicates the type of the next Extension header if more Extension Headers are in the packet. The Next Header field of the last Extension header indicates the upper layer protocol (such as TCP, UDP, or ICMPv6) contained within the packet.

Figure 12-2 illustrates the IPv6 Header with zero or more Extension Headers. They form a chain of headers. Each Header indicates in its Next Header field the type of Header that comes after it, until the last header in the chain identifies the upper layer protocol.

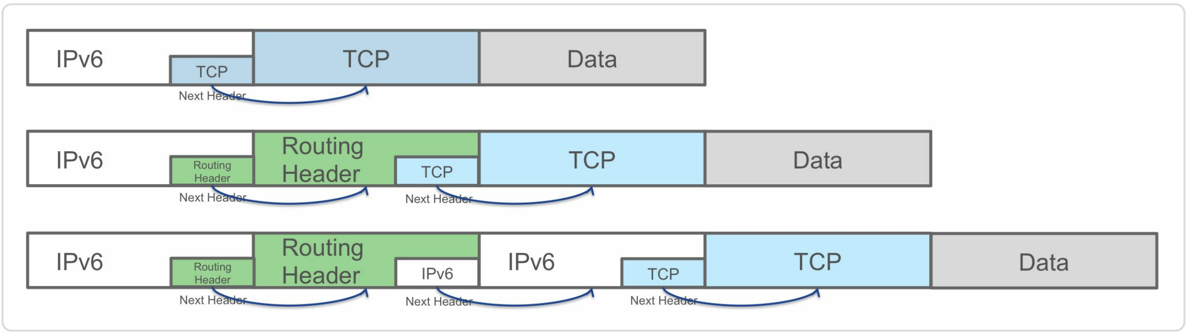


Figure 12-2: Chaining IPv6 headers

12.4 Routing Header

One of the IPv6 Extension Header types is the Routing Header; its type number is 43. The Routing Header is used by an IPv6 source to list one or more intermediate nodes for the packet to travel to on its path to the final destination. The source node can use the Routing Header to source route the packet through the network. Different variants of Routing Headers have been defined: Source Route (Deprecated IETF RFC 5095), Mobility support (IETF RFC 6275), Routing Protocol for Low-Power and Lossy Networks (RPL) (IETF RFC 6554).

The Routing Header has the format (IETF RFC 2460) shown in Figure 12-3.

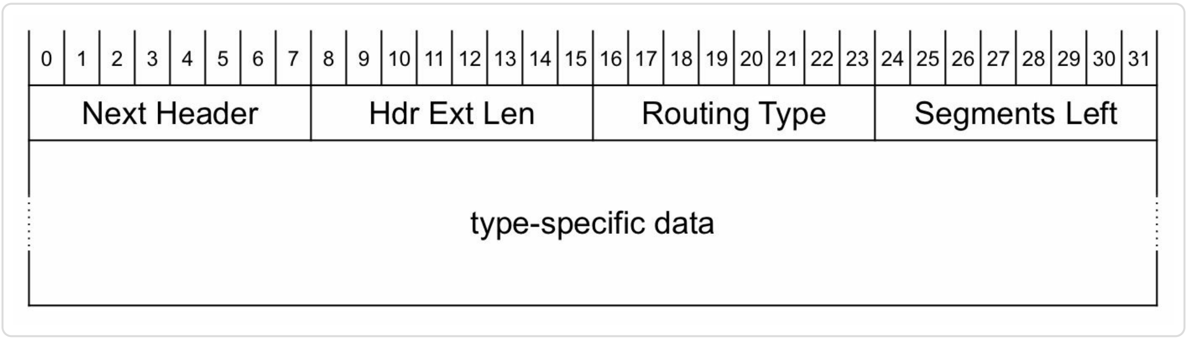


Figure 12-3: IPv6 Routing Header format

- Next Header: Identifies the type of header immediately following the Routing header. Uses the same values as the IPv4 Protocol field.
- Hdr Ext Len: Length of the Routing Header.
- Routing Type: identifier of a particular Routing Header type.
- Segments Left: Number of route segments remaining, i.e. the number of explicitly listed intermediate nodes still to be visited before reaching the final destination.

- type-specific data: determined by the Routing Type field.

In general, Extension Headers are not examined or processed by intermediate nodes along the path of the packet to its destination. The intermediate nodes forward the packet based on the Destination Address in the main IPv6 Header. The exception is the Hop-by-hop Options Header that conveys optional information that must be examined by every node along the path of the packet. At the destination node, the next header, as indicated in the Next Header field of the IPv6 Header, is examined and processed. The next header can be an Extension Header or an upper-layer protocol header.

If a node receives a packet and the Destination Address of the packet corresponds to an address of the node, then the packet examines the Extension Header, if present. If the Extension Header is a Routing Header with a Routing Type that the node does not recognize, then the behavior of the node depends on the value of the Segments Left field:

- If the Segments Left field value is zero, then the node ignores the Routing Header and processes the next header in the packet.
- If the Segments Left field value is not zero, then the node discards the packet and sends an ICMP "Parameter Problem" message to the Source Address of the packet.

IETF RFC 2460 defines the Routing Header variant with Routing Type 0, also known as "RH0". This type of Routing Header defines the classic source routing model for IPv6. The type 0 Routing Header has been deprecated by IETF RFC 5095, motivated by security concerns. For more details about the security threats, see IETF RFC 5095 and RFC 4942. These security concerns have been addressed in the SRH specification.

12.5 Segment Routing Header

The Segment Routing Header (SRH) is a new type of the existing Routing Header, with a proposed Routing Type 4.

The SRH format is very similar to the Routing Type 0 Routing Header. The security concerns that lead to the depreciation of the Routing Type Routing Header are addressed for the Segment Routing Header through the HMAC TLV that can be added in the header.

The Segment Routing Header inherits the Routing Header properties: it should only appear once in the packets; and if the Segments Left field has a value of 0 then the SRH is ignored (not dropped) and the packet is processed based on the next header in the packet.

Figure 12-4 shows the format of the Segment Routing Header (SRH).

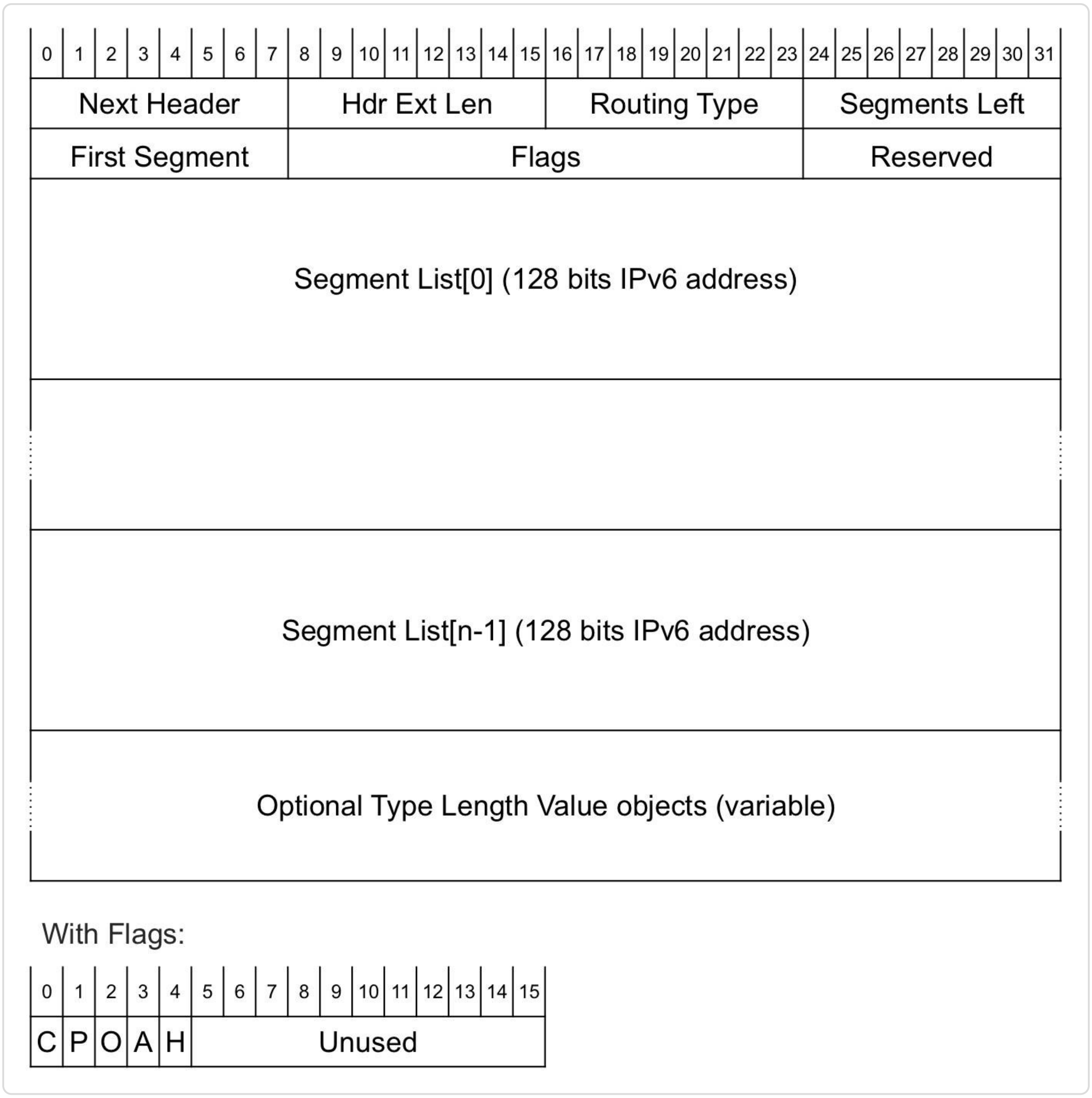


Figure 12-4: Segment Routing Header format

- Next Header: Identifies the type of header that follows the SRH.
- Hdr Ext Len: the length of the SRH header.
- Routing Type: proposed value for the SRH type: 4.
- Segments Left: index of the current Active Segment in the Segment List of the SRH. This index is decremented for each completed segment.
- First Segment: index of the first segment of the path in the Segment

List of the SRH. Since the elements in the Segment List are placed in reverse order, this is the last element of the Segment List. This helps to access the TLV's following the SID list.

- Flags:
  - C-flag (Clean-up): If set, then the SRH must be removed from the packet prior to forwarding the packet on the last segment; this is similar equivalent to MPLS PHP.
  - P-flag (Protected): Set when a Segment Endpoint Node has rerouted the packet through FRR mechanism.
  - O-flag (OAM): If set, then this packet is an Operations And Management (OAM) packet.
  - A-flag (Alert): If set, then important Type Length Value (TLV) objects are present in the SRH.
  - H-flag (HMAC): If set, then the HMAC TLV is present and is encoded as the last TLV of the SRH.
  - Unused: for future use.
- Reserved: for future use.
- Segment List[x]: List of 128 bit entries, representing each segment of the path. The Segment List is encoded in the reverse order of the path: the last segment is in the first position (Segment List[0]) of the list and the first segment is in the last position (Segment List[n-1]).
- Optional Type/Length/Value objects, see further.

HIGHLIGHT

The active SID is in the destination address (DA) field of the IPv6 header.

When a node receives a packet with the destination address equal to a local SID, we say that the node acts as a segment endpoint or SID endpoint.

A SID endpoint performs the instruction/function associated with the SID: e.g. null function for a Prefix-SID, forward on a specific interface for a (global) Adj-SID.

A SID endpoint looks in the SRH and finds the next SID to activate. It uses the Segments Left field as an offset in the SID list.

Activating a SID means copying that SID in the destination address and forwarding the packet according to this updated destination address.

If the SID is the last SID (Segments Left = 0), then the SID endpoint processes the payload according to the Next Header field in the SRH.

The optional TLVs in the SRH contain information that may be used by the Segment Endpoint Node, the node that corresponds to the IPv6 Destination Address of the packet. The primary use of these TLVs is to provide information to the final destination of the packet about the path of the packet. The routing layer does not use information in the TLVs, but it can be used for other purposes, such as OAM.

Ingress Node TLV

The Ingress Node TLV is optional and contains a 128-bit field with the identity of the node where the packet entered the Segment Routing Domain.

Egress Node TLV

The Egress Node TLV is optional and contains a 128-bit field with the identity of the node where the packet is expected to exit the Segment Routing Domain.

Opaque Container TLV

The Opaque Container TLV is optional and contains 128 bits of data. This data is not relevant for the routing layer, but can be used to convey other information to the destination node of the packet.

Padding TLV

The Padding TLV is optional and is used to align the SRH on an 8 octet boundary.

HMAC TLV

HMAC TLV is optional and contains the HMAC information. The format of the HMAC TLV is shown in Figure 12-5.

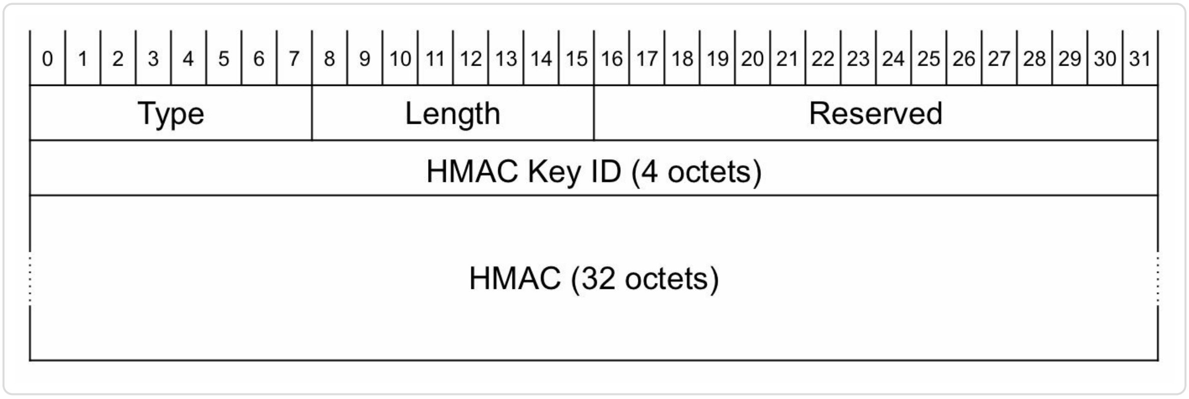


Figure 12-5: HMAC TLV format

12.6 The SRH Procedures

A packet with an SRH traverses various nodes in the network. This section explains how the different nodes on the path process the packet. The network topology in Figure 12-6 illustrates the path of a packet with SRH. A packet is steered from Source Node Node11 via Segment Endpoint Nodes Node1 and Node6 to Destination Node12. Node12 is also Segment Endpoint Node of the last segment.

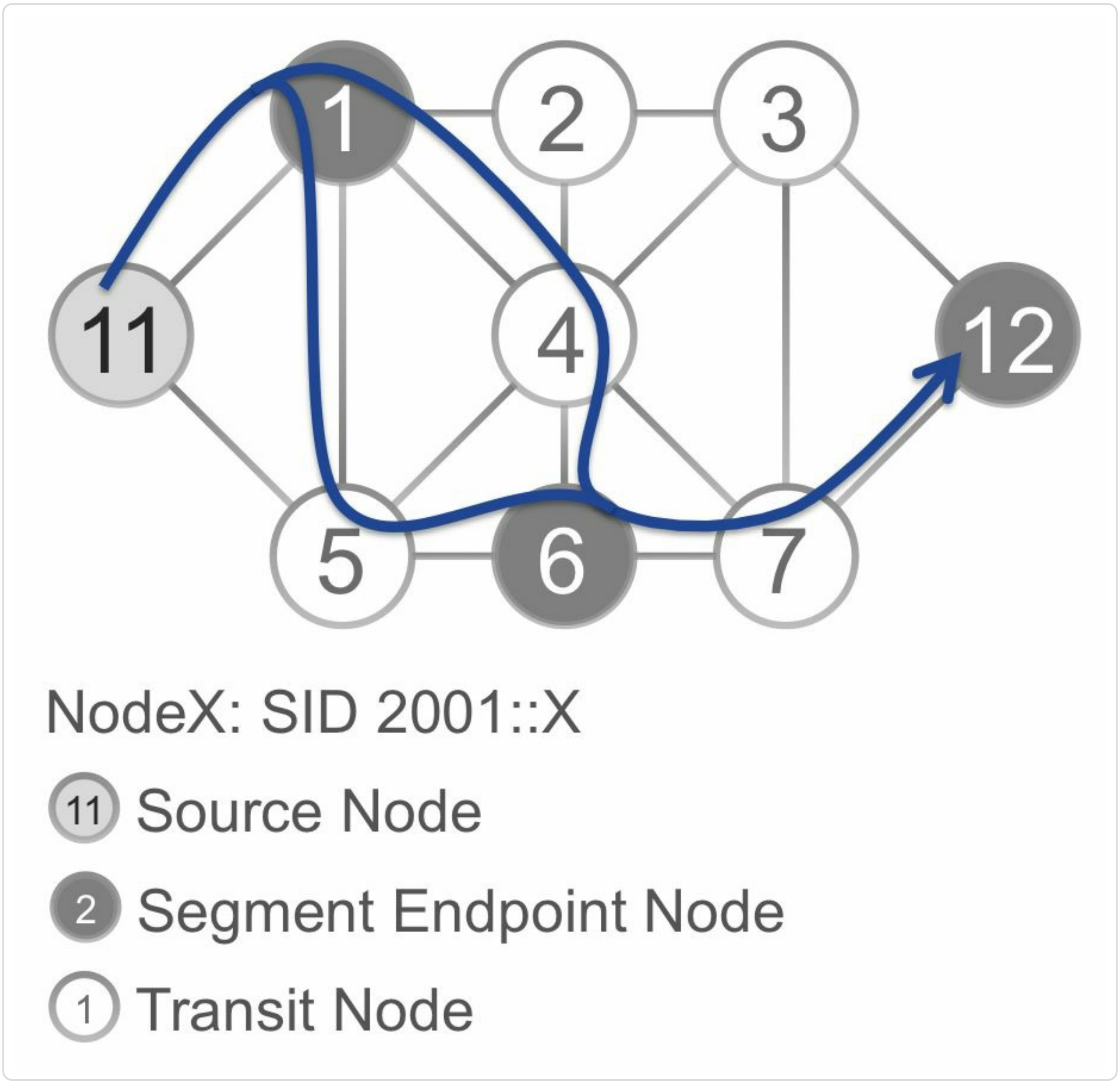


Figure 12-6: SRH Procedures – network topology

12.6.1 Source Node

A Source Node can be any node originating an IPv6 packet with a Segment Routing Header:

- A host originating an IPv6 packet with SRH
- An ingress node (in many cases a router) of a SRv6 Domain that inserts a SRH or pushes a new outer IPv6 Header with SRH.

The Segment List specifies the path of the packet. This path can be locally configured, locally computed, or provided by an external controller. The Segment List is {S[0], S[1], …, S[n-1]}, with S[x-1] the x^{th} segment in the list. The Source Node inserts the Segment List in the SRH in the packet header.

When the Source Node originates the packet it creates the SRH for the packet as follows and inserts it in the header:

- "Segment List": n segments encoded in the reverse order of the path (last segment in first position, first segment in last position): (0:S[n-1], 1:S[n-2], …, n-1:S[0]), where "x:" indicates the index in the SRH.
- "Segments Left" field: set to n-1
- "First Segment" field: set to n-1
- The DA of the packet is set as the first segment of the path
  - DA = S["Segments Left"]
- HMAC TLV and/or other TLVs may be added to the SRH

The Source Node sends the packet out to the IPv6 Destination Address of the packet.

Node11 in Figure 12-6 is the Source Node of a packet with Segment List {SID(1), SID(6), SID(12)}. The Segment List field in the SRH is encoded as L=(0:SID(12), 1:SID(6), 2:SID(1)); L is represented in the order as found in the SRH. The Segments Left field and First Segment field are set to 2. The Destination Address is set to L[2]=SID(1); this is the Active Segment. The packet with SRH is represented in Figure 12-7 between Node11 and Node1.

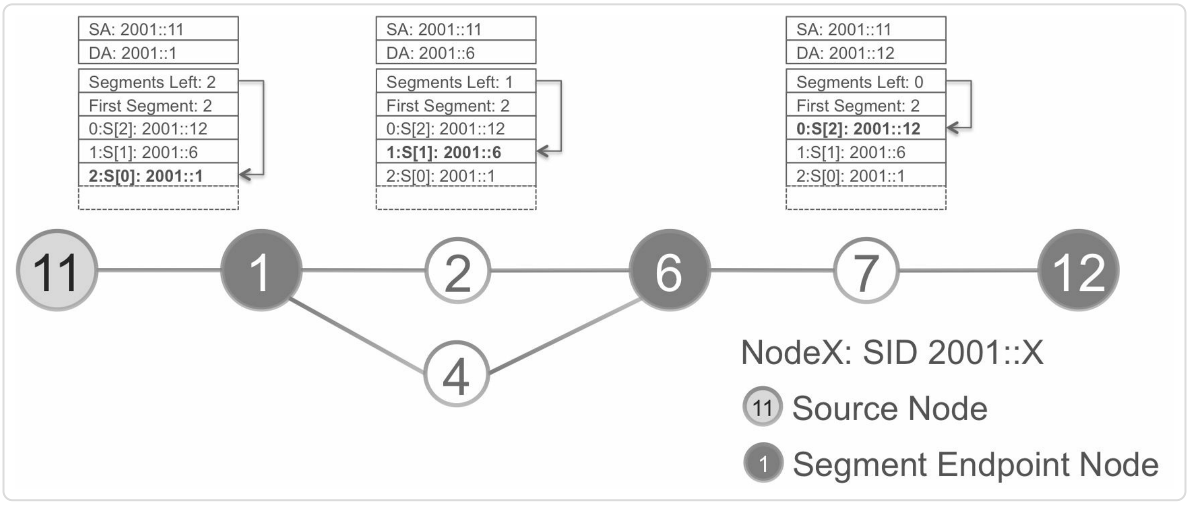
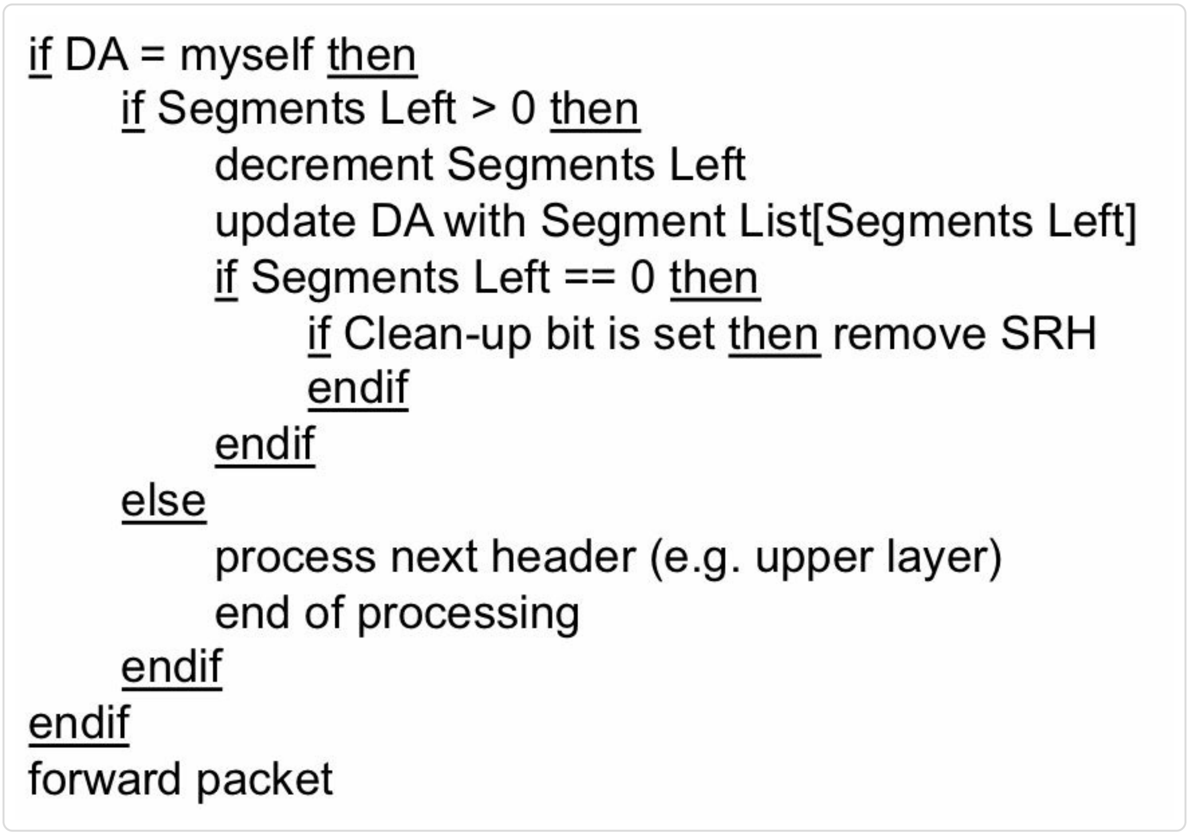


Figure 12-7: SRH Procedures – Packet Header

12.6.2 Segment Endpoint Node

A Segment Endpoint Node is a node whose address is the Destination Address of an IPv6 packet with SRH. The Segment Endpoint Node does the operations shown in Figure 12-8 when receiving a packet with SRH.



```
if DA = myself then
    if Segments Left > 0 then
        decrement Segments Left
        update DA with Segment List[Segments Left]
        if Segments Left == 0 then
            if Clean-up bit is set then remove SRH
            endif
        endif
    else
        process next header (e.g. upper layer)
        end of processing
    endif
endif
forward packet
```

Figure 12-8: Segment Endpoint Node operations

The Active Segment is the segment in the Segment List pointed to by the Segments Left field. The Active Segment is also set as the Destination Address of the packet. At each Segment Endpoint Node the Destination Address is updated with the next Active Segment found in the Segment List. This is compliant with IETF RFC 2460 rules for the Routing Header.

Node1 in Figure 12-6 is the Segment Endpoint Node of the first segment. Node1 finds Segments Left = 2 in the packet. It decrements the Segments Left field to 1 and updates the IPv6 Destination Address to the next segment in the list: 2001::6. This is L[Segments Left] = L[1] = 2001::6. Node1 sends the packet to Destination Address 2001::6. Node1 load-balances packets over the two paths towards Node6.

When the packet arrives at Node6, Node6 applies the same procedures. It decrements the Segments Left field to 0 and updates the IPv6 Destination Address to L[Segments Left] = L[0] = 2001::12, and sends the packet to 2001::12.

Node12 is the final destination of the packet. The packet arrives on Node12 with Segments Left field 0, hence Node12 processes the higher layer protocol in the packet.

12.6.3 Transit Node

A Transit Node is a node on the path of the packet, but that is not the endpoint of a segment. IETF RFC 2460 specifies that a node may not process the Routing Header if that node is not corresponding to the Destination Address of the packet. The IPv6 specification indicates that such Transit Node must forward the packet to its IPv6 Destination Address without touching the SRH. A Transit Node does not need to support SRv6 to transport the packet.

Node2, Node4, and Node7 in Figure 12-6 are Transit Nodes. They do not examine the SRH and do not update the Destination Address in the IPv6 Header. They simply forward the packet based on its IPv6 Destination Address.

12.7 SRH Insertion versus Pushing IPv6 Header

SRv6 allows multiple modes of operation:

- SRH insertion at the source of the packet
- SRH insertion at the ingress node/router of the Segment Routing Domain
- Push a new outer IPv6 Header with SRH at the ingress node of the Segment Routing Domain

These modes of operation are illustrated with the network topology in Figure 12-9. The source of the IPv6 packet is NodeA, its destination is NodeB. The packet follows the path via Node2.

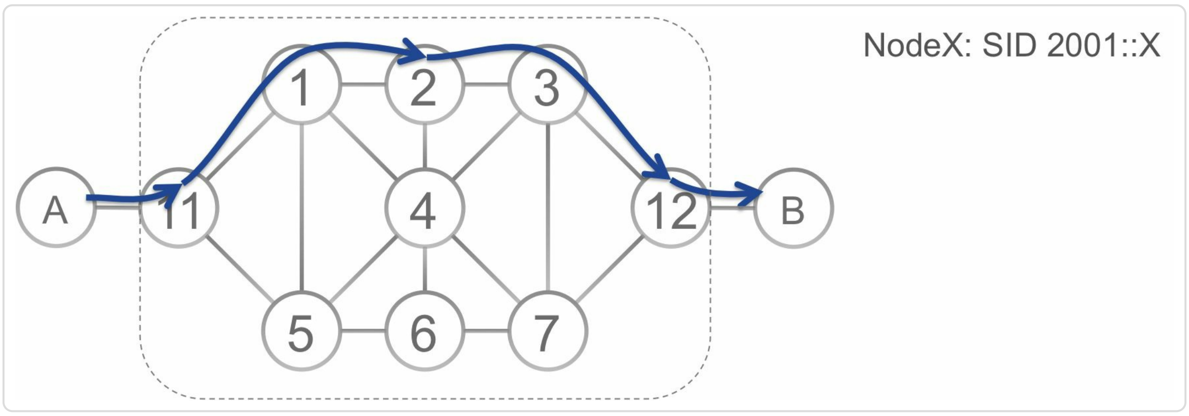


Figure 12-9: SRH insertion versus Pushing IPv6 Header – network topology

12.7.1 SRH Insertion at Source

The SRv6 capable source node originates the IPv6 packet with a SRH. The SRH stays on the packet and is used along the path of the packet. Finally the packet is delivered to its destination with the SRH in the header. See Figure 12-10.

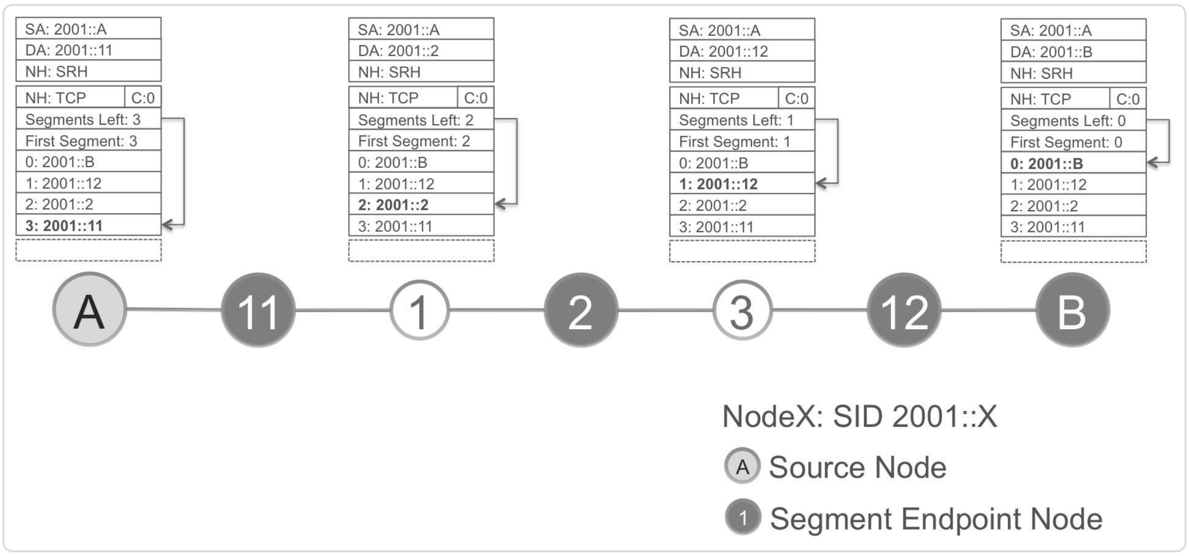


Figure 12-10: SRH insertion at the source of the packet

Optionally, the SRH may be removed from the packet prior to delivering it to its destination. This can be achieved by setting the C-flag (Clear) in the SRH. See Figure 12-11. Node12 receives the packet with Segments Left 1 and C-flag set. Node12 applies the operations in Figure 12-8. Node12 first decrements the Segments Left field, which now becomes 0. Then Node12 updates Destination Address to L[Segments Left] = L[0] = 2001::B. Since Segments Left == 0 and C-flag is set, Node12 removes the SRH from the header and forwards to packet to its destination NodeB.

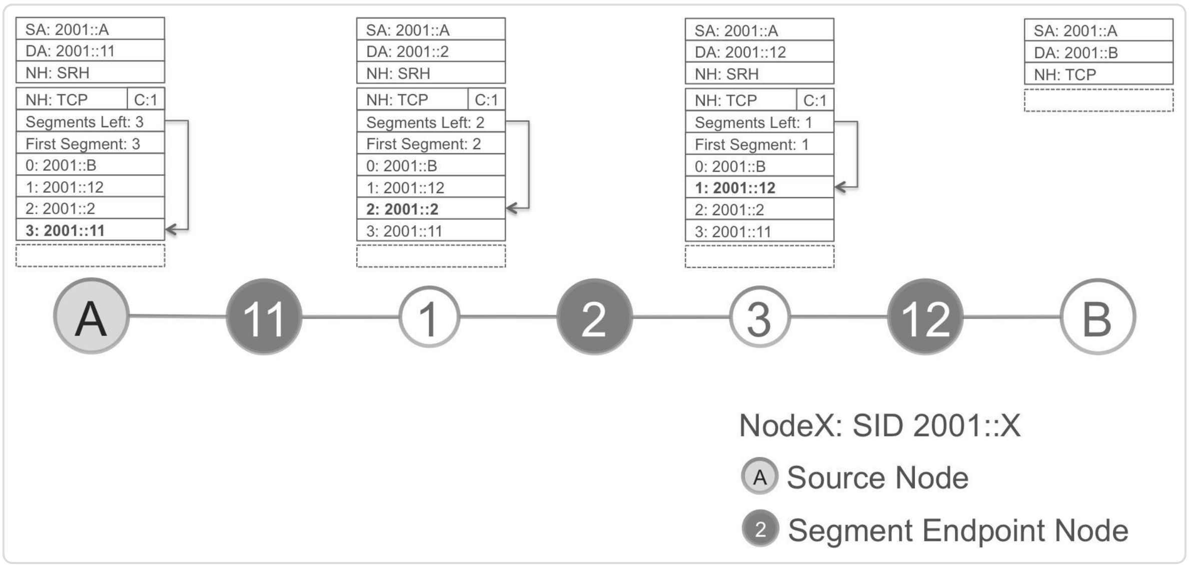


Figure 12-11: SRH insertion at the source of the packet, C-flag set

12.7.2 SRH Insertion at Ingress

The non-SRv6 capable source node originates an IPv6 packet without a SRH. When the packet arrives at the ingress node of the Segment Routing Domain, the ingress node inserts a SRH in the packet. The SRH is removed from the packet prior to delivering it to its destination; this is achieved by setting the C-flag (Clear) in the SRH. See Figure 12-12. Node12 receives the packet with Segments Left 1 and C-flag set. Node12 applies the operations in Figure 12-8. Node12 first decrements the Segments Left field, which now becomes 0. Then Node12 updates Destination Address to L[Segments Left] = L[0] = 2001::B. Since Segments Left == 0 and C-flag is set, Node12 removes the SRH from the header and forwards to packet to its destination NodeB.

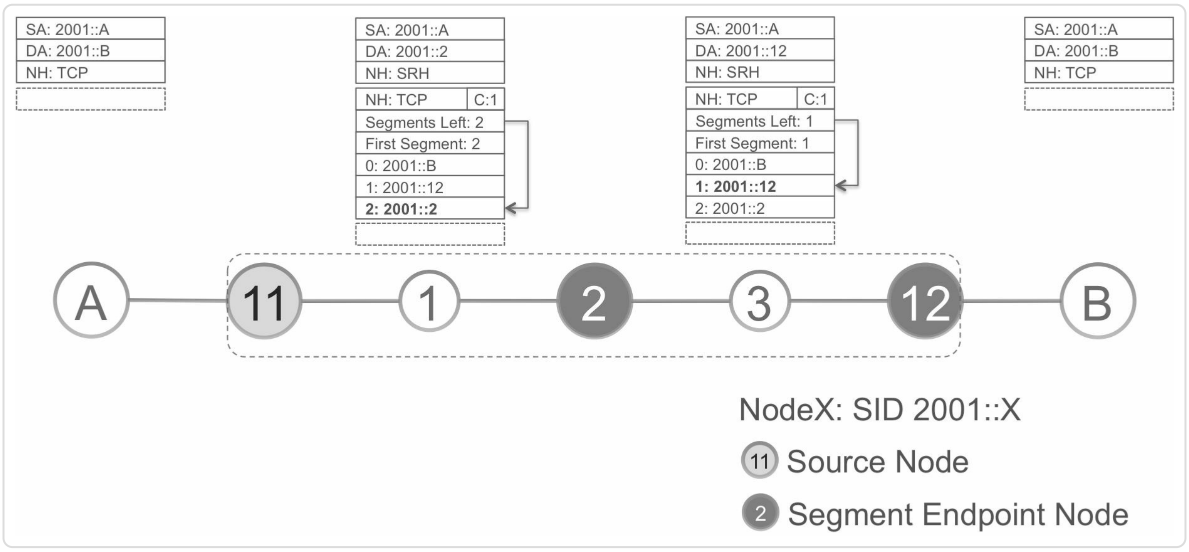


Figure 12-12: SRH insertion at the ingress node of the SR Domain

12.7.3 Encap Header Insertion at Ingress

A non-SRv6 capable source node originates an IPv6 packet without a SRH. When the packet arrives at the ingress node (typically a router) of the Segment Routing Domain, it pushes a new outer IPv6 Header with SRH on the packet. The complete outer IPv6 Header with SRH is removed at the egress node (typically a router) of the Segment Routing Domain. See Figure 12-13. Node12 is the nodes corresponding to the Destination Address in the outer IPv6 header. Therefore, Node12 processes the next header, which is an IPv6 Header. Node12 strips the outer header from the packet and forwards the exposed IPv6 packet to its destination NodeB.

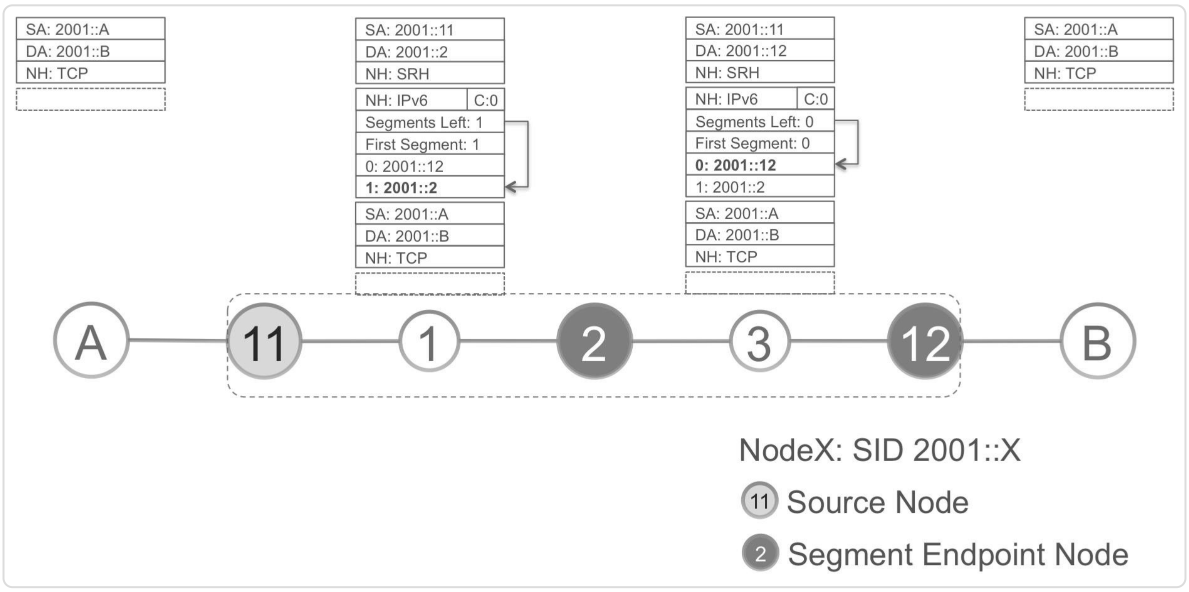


Figure 12-13: Push outer IPv6 Header with SRH at the ingress node of the SR Domain

12.8 Summary

- SRv6 is the implementation of the Segment Routing architecture using IPv6 data plane
- Simpler because no MPLS signaling or forwarding, better scale with summarization
- Segment Routing for the IPv6 data plane (SRv6) specifies a Segment Routing Header (SRH) that adheres to the Routing Header specification of the IPv6 standard
- A SRv6 Segment ID is an IPv6 address and Segment List is encoded in the SRH
- SRH can be added by the node that originates packets or by the ingress node of a network
- SRv6 interworks with standard IPv6 nodes; only specific hosts and routers that operate on the SRH need to support SRv6
- Features like TI-LFA protection, Traffic Engineering, etc. also extend to SRv6

12.9 References

- [draft-ietf-6man-segment-routing-header] Previdi, S., Filsfils, C., Field, B., Leung, I., Linkova, J., Kosugi, T., Vyncke, E., and D. Lebrun, “IPv6 Segment Routing Header (SRH)”, draft-ietf-6man-segment-routing-header (work in progress), September 2016, https://datatracker.ietf.org/doc/draft-ietf-6man-segment-routing-header.
- [RFC2113] Katz, D., “IP Router Alert Option”, RFC 2113, DOI 10.17487/RFC2113, February 1997, https://datatracker.ietf.org/doc/rfc2113.
- [RFC2460] Deering, S. and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460, DOI 10.17487/RFC2460, December 1998, https://datatracker.ietf.org/doc/rfc2460.
- [RFC4942] Davies, E., Krishnan, S., and P. Savola, “IPv6 Transition/Co-existence Security Considerations”, RFC 4942, DOI 10.17487/RFC4942, September 2007, https://datatracker.ietf.org/doc/rfc4942.
- [RFC5095] Abley, J., Savola, P., and G. Neville-Neil, “Deprecation of Type 0 Routing Headers in IPv6”, RFC 5095, DOI 10.17487/RFC5095, December 2007, https://datatracker.ietf.org/doc/rfc5095.

SUMMARY

The purpose of this first part of the book on Segment Routing has been to provide an objective description of its fundamental block along with some subjective aspects and context behind its development. We believe that Segment Routing will influence the evolution of networks in a significant way in the years to come, much like how MPLS did in the past. The consensus within the networking industry and the rapid acceptance of this technology is perhaps the strongest testament for this.

Segment Routing is constantly evolving with the participation of a large community of network operators, network engineers and developers of networking platforms. We would like to encourage the reader to stay abreast of all these technological developments through the various IETF working groups involved in this process as well as other forums and resources on the Internet. We would also like to point to the http://www.segment-routing.net webpage, which is updated by a community of SR technology developers.

In summary, the following key takeaways could be extracted from this book:

- The Segment Routing Architecture seeks the right balance between distributed intelligence (control plane on routers) and centralized optimization & programming (via controllers) to become the foundation for Software Defined Networks.
- Segment Routing reduces the state in the network to Segments (e.g. Prefix Segments) while the path through the network is carried in the

packet in the form of Segment List. This allows the network to scale for handling large number of application specific flows and their paths through the network without the need for handling their individual state, which is now in the packet.

- Segment Routing technology is developed using extensions to existing network protocols (e.g. OSPF, ISIS, BGP, etc.) and does not introduce any new protocol. It in fact simplifies the MPLS network operations by eliminating protocols like LDP and RSVP-TE in many network designs.
- Segment Routing works for both IPv4 and IPv6 over MPLS data plane for as well as native IPv6 data plane (called SRv6).
- Segment Routing leverages the existing MPLS data plane and interworks with existing MPLS control plane protocols thereby allowing deployment in existing IP/MPLS networks with minimal (if any) service disruption. The services can be seamlessly migrated from existing transport to SR and the base functionality is mostly a software upgrade for most routers.
- TI-LFA, enabled by Segment Routing, provides automated backup paths over the post-convergence path in any topology to enable sub 50ms protection for all services in the network. It is simple to provision and also protects IP and LDP traffic.
- Segment Routing addresses day one IP routing problems like microloops.
- Segment Routing is applicable for all types of networks from Service Provider Core/Access/Aggregation, Data Centers, Internet Peering and Enterprise networks.
- Segment Routing enables end-to-end Traffic Engineering capabilities

for applications and services even across massively scaled multi-domain networks

This book is the first part of a series. The next part of the series will detail the Traffic Engineering solution, the SDN/controller interaction with the SR infrastructure, SR on the host and SRv6.

We would also like to encourage the readers to connect via our blog http://segment-routing-book.blogspot.com where they can submit feedback and comments as well as access additional and updated resources related to this book such as configuration examples. While Segment Routing evolves, this blog would be our attempt to keep the content alive between versions and notify readers of its revisions.

The authors of the book can also be reached via email segment-routing-book@googlegroups.com to send errata, comments, or suggestions.