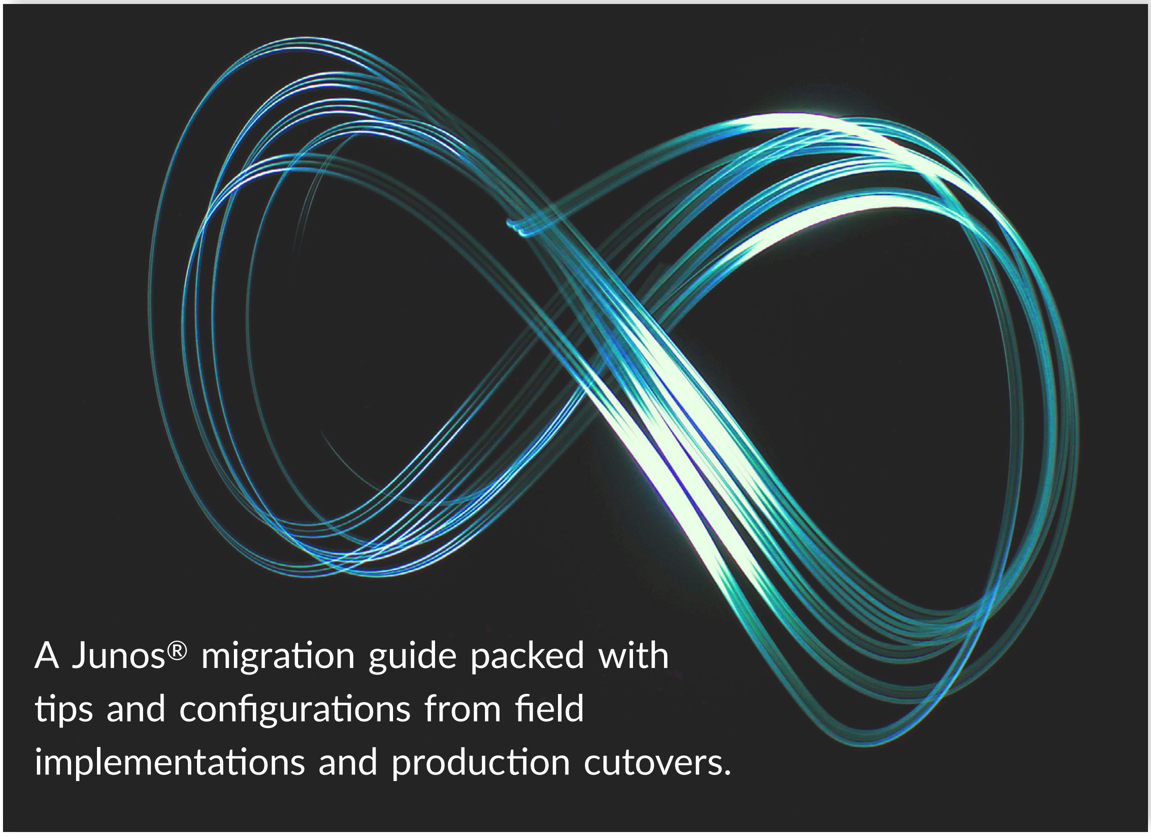
Engineering Simplicity

DAY ONE: MIGRATING TO SEGMENT ROUTING


A Junos® migration guide packed with tips and configurations from field implementations and production cutovers.

By Shraddha Hegde, Chris Bowers, and Melchior Aelmans

DAY ONE: MIGRATING TO SEGMENT ROUTING

Recent activity in IETF and the network community suggest using the Segment Routing (SR) control plane for large transport networks. That's because RSVP and LDP can't address all of the challenges operators of large scale modern transport networks face. Hence the development of Segment Routing architecture that not only addresses these challenges but can greatly simplify operations by reducing the variety of protocols needed. The authors point out other cool advantages of SR along the way, too, as they take you through a Junos migration from RSVP to SR in a speedy but thorough style. The Appendix offers two alternative approaches to solving SR migration scenarios plus all the configurations from the book's lab.

"Hegde, Bowers, and Aelmans lead you through the ins and outs of a Junos migration to Segment Routing with ease. Their world class knowledge, both theoretical and practical, is a boon to the reader as they add field-derived insights to each migration stage. I'm ecstatic they've written this needed Junos book, and trust me, you're in good hands the entire migration."

John Scudder, IETF Routing Area Director, Juniper Networks Distinguished Engineer

"This is a must-read book for every practitioner focused on using Segment Routing to address the challenges in provisioning LSPs and distribution of labels in large-scale transport networks. The book is unique as it focuses on providing concrete suggestions, tips, and tricks with detailed practical, hands-on examples and case studies. My kudos to the authors for an excellent job in delivering useful content." ***Raj Yavatkar, PhD, Chief Technology Officer, Juniper Networks***

"Migrating an existing MPLS network in-service to Segment Routing has a lot of larger and smaller obstacles, many of them unseen at the first glance. Whether you come from ISIS or OSPF, whether you are using LDP or RSVP, this book shows you what you need to think about, which options you have and how to do all of this with Junos. Knowing the authors from several years of common work at the IETF I can assure you they made this book as efficient and helpful for the task as it can be."

- ***Dr. Martin Horneffer, Squad Lead Internet Backbone Architecture, Tier 1 Operator***

IT'S DAY ONE AND YOU HAVE A JOB TO DO:

- Understand how to migrate RSVP to the segment routing control plane.
- Make the needed configuration changes on routers running the Junos® OS.
- Verify segment routing operations on Junos devices.
- Use TI-LFA to replace RSVP-TE.
- Two alternative migration scenarios authored by Colby Barth and Jordan Stewart.

ISBN 978-1736316030

53000

9 781736 316030

JUNIPer NETWORKS

Day One: Migrating to Segment Routing

by Shraddha Hegde, Chris Bowers, and Melchior Aelmans

JUNIPer
NETWORKS

Published by Juniper Networks Books
Authors: Shraddha Hegde, Chris Bowers, and Melchior Aelmans
Technical Reviewers: Ron Bonica, Julian Lucek, Colby Barth
Editor in Chief: Patrick Ames
Copyeditor: Nancy Koerbel

About the Authors
Shraddha Hegde is a Principal Engineer in Juniper Networks' Routing Protocols Group. She (co-)authored five IETF RFC documents and several internet drafts. Shraddha has 20 years of experience in Networking Domain in the area of IGP, FRR, Segment Routing, MPLS, and IP Security. She has vast experience in building and deploying highly scalable data-communications platforms. Prior to joining Juniper, Shraddha served as a Systems Architect for a major telecom equipment vendor.

Chris Bowers is a Principal Engineer in Juniper Networks' Routing Protocols Group. He has co-authored seven IETF RFCs and co-chaired the IETF Routing Area Working Group. Chris has a broad range of networking experience from optical networking, security, and IP/MPLS technologies. Recently his focus has been on using segment routing for fast-reroute and micro-loop avoidance applications, as well as scaling of IGP implementations and deployments.

Melchior Aelmans is a Consulting Engineer at Juniper Networks, where he has been working with many operators on the design and evolution of their networks. He has over 15 years of experience in various operations and engineering positions with Cloud Providers, Data Centers, and Service Providers. Melchior enjoys evangelizing and discussing routing protocols, routing security, internet routing and peering, and data center architectures. He also participates in IETF and RIPE, is a regular attendee and presenter at conferences and meetings, is a member of the NANOG Program Committee, and a board member at the NLNOG Foundation.

Author Acknowledgments
The authors would like to thank, in random order, Patrick Ames, Ron Bonica, Colby Barth, Julian Lucek, and John Scudder (Juniper Networks) for their valuable contributions, content, input, review, comments, and emotional support.

In particular the authors would like to thank the authors of Appendix B and C, Colby Barth and Jordan Stewart, for providing their insights and allowing us to include their work in this book.

Version History: v1, June. 2021
2 3 4 5 6 7 8 9 10
Comments, errata: dayone@juniper.net

Welcome to Day One

This book is part of the *Day One* library, produced and published by Juniper Networks Books. *Day One* books cover the Junos OS and Juniper Networks network administration with straightforward explanations, step-by-step instructions, and practical examples that are easy to follow.

- Download a free PDF edition at https://www.juniper.net/dayone
- Purchase the paper edition at Vervante Corporation (www.vervante.com).

Key Segment Routing Resources

While writing this book the authors were greatly influenced by existing work previously completed by other authors. We'd like to thank and recognize the authors of the following works and encourage readers to explore them:

- *Day One: Configuring Segment Routing with Junos* (Julian Lucek and Krzysztof Szarkowicz) https://www.juniper.net/documentation/en_US/day-one-books/DO_SegmentRouting.pdf
- *Day One: Inside Segment Routing* (Anurag Khare and Colby Barth) https://www.juniper.net/documentation/en_US/day-one-books/DO_InsideSR.zip

A What You Need to Know Before Reading This Book

Before reading this book, you need to be familiar with the basic administrative functions of the Junos operating system, including the ability to work with operational commands and to read, understand, and change Junos configurations.

There are several books in the *Day One* library on learning the Junos OS, at http://www.juniper.net/dayone. This book makes a few assumptions about you, the reader:

- You are familiar with and versed in using the Junos CLI for router configuration.
- You understand how existing Multiprotocol Label Switching (MPLS) networks are designed and configured, and how they function. This includes a basic understanding of OSPF, ISIS, BGP, LDP, and RSVP-TE.
- You can build out the lab topologies used in this book without detailed setup instructions.

What You Will Learn by Reading This Book

After reading this book you will be able to:

- Understand how to migrate RSVP to the segment routing control plane.
- Make the needed configuration changes on routers running the Junos OS.
- Verify segment routing operations on Junos devices.
- Use TI-LFA to replace RSVP-TE.

Introduction

Since the initial publication of the drafts resulting in RFC2205 (RSVP) and RFC3036 (LDP), RSVP and LDP have been the default choices of implementers for large wide area networks to distribute labels in MPLS (RFC3031) networks.

In late 2001 RFC3209 was published in order to define RSVP Traffic Engineering (RSVP-TE), which allowed RSVP to "support the instantiation of explicitly routed LSPs, with or without resource reservations. It also supports smooth rerouting of LSPs, preemption, and loop detection" (source RFC3209). Many additional RFCs have been published over time to tailor RSVP to operators' needs and demands.

Recent activity in IETF and with implementers suggests using Segment Routing (SR) technologies for large transport networks. The main challenge with large networks is distribution and storage of labels, in particular when provisioning backup Label Switched Paths (LSPs) throughout, as network routers can get saturated with these transit states.

RSVP and LDP don't, or partly don't, address some of the challenges operators of large scale transport networks face in today's networks.

Hence a new routing architecture, SR, was developed to not only address current issues, but perhaps even more importantly, to reduce the variety of protocols used in networks and thus simplify operations.

While the arguments presented in this book are legitimate reasons not to use RSVP or LDP protocols for the signaling path throughout the network, and they show that there are other options available, readers might (incorrectly) conclude the authors do not believe RSVP or LDP should be used as the label distribution protocol for a transport network. This is, however, not the case. Ultimately, it is up to individual operators to decide which protocol is "the best" for their application, a decision that should be based on business and operational, as well as technical, reasons.

The authors assume the reader has in-depth knowledge of the technologies (RSVP, LDP, MPLS, and Segment Routing) used in this book and expect the reader to also have read the *Day One* books cited in *Key Segment Routing Resources*.

This book isn't intended as a comprehensive introduction to Segment Routing. Instead, it is more of pure migration guide, containing suggestions, tips, and tricks taken from customer conversations and field trials.

The authors also suggest readers examine Appendixes B and C, as they provide an alternate approach to solving SR migration scenarios.

Terminology

Throughout this book you will come across terms and abbreviations that may be new to you. Here is how the authors use these terms.

Anycast-SID

An anycast-SID (a type of prefix-SID) corresponds to a prefix that may be advertised by multiple nodes. Paths constructed using anycast-SIDs can have useful resiliency properties.

Node-SID

A node-SID (a type of prefix-SID) corresponds to a prefix that should only be advertised by a single node. Knowing that a prefix should only be advertised by a single node is useful when doing computations to construct paths using prefix-SIDs.

Prefix-SID

A prefix-SID (prefix segment identifier (SID)) is associated with an IP prefix. The prefix-SID is manually configured from the segment routing global block (SRGB) range of labels.

SRGB

SRGB is the set of global segments in the SR domain. If a node participates in multiple SR domains there is one SRGB for each SR domain. In SR-MPLS, SRGB is a local property of a node and identifies the set of local labels reserved for global segments. In SR-MPLS, using identical SRGBs on all nodes within the SR domain is strongly recommended. Doing so eases operations and troubleshooting as the same label represents the same global segment at each node. In SRv6, the SRGB is the set of global SRv6 SIDs in the SR domain.

TI-LFA

Topology Independent Loop-Free Alternates (TI-LFA) is a method of computing and constructing fast-reroute backup paths using SR-MPLS labels. The TI-LFA backup path for a given destination prefix will correspond to the path that will eventually be taken by the traffic after the IGP converges (the post-convergence path). It is referred to as "topology-independent" because, with enough SR-MPLS labels, the post-convergence path can always be constructed, independent of network topology.

Chapter 1

Meet the Legacy Network

This book uses a single topology to discuss different migration scenarios and to clarify examples, configurations and show commands. This chapter introduces you to this baseline topology. We tried to come somewhat close to an actual network, but keep in mind that every network differs so whatever you read in this book…. *don't try it in a live network without thorough testing!*

The brownfield network shown in Figure 1.1 consists of eight nodes initially running OSPF to provide underlay connectivity. In the *Day One* lab for this book we used Juniper MX routers, and over the course of this book we illustrate how this network can migrate from a plain vanilla OSPF to OSPF-SR, and then to ISIS-SR. As stated earlier, at the beginning of the Chapter 2 the network runs OSPF as a starting point.

In all the scenarios you migrate to, you will leverage different control plane protocols. What they all have in common is they all use MPLS as the data plane.

There are several observations of note about the initial configuration of the eight-node network in Figure 1.1:

- The network consists of eight Juniper Networks MX routers.
- All routers are running Junos OS 20.2R1, which supports OSPF-SR and ISIS-SR.

MORE? For more about Junos SR support see: https://www.juniper.net/documentation/en_US/junos/topics/concept/source-packet-routing.html.

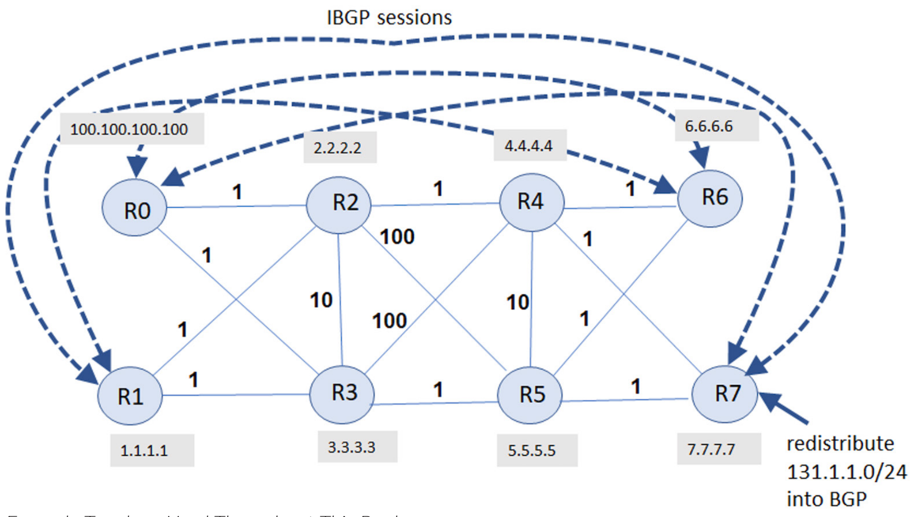


Figure 1.1 Example Topology Used Throughout This Book

- A plain vanilla OSPF is initially used as the IGP. Throughout the book we will introduce OSPF-SR and ISIS-SR.
- LDP is used to distribute the MPLS labels, which are used for forwarding.
- The IGP metrics configured on the links are shown in Figure 1.1.
- R0, R1, R6, and R7 function as PE nodes. There is a full mesh of IBGP sessions between the four PE nodes.
- R2, R3, R4, and R5 are P routers. They only perform MPLS forwarding, hence they do not need to run BGP or install any of the BGP routes known to R0, R1, R6, and R7.
- In order to keep configurations in the examples 'simple', the service traffic transported across the MPLS core corresponds to IPv4 service prefixes. It is straightforward to extend these examples to various flavors of VPN service prefixes.
- At R7, the prefix 131.1.1.0/24 is being redistributed into BGP. The prefix 131.1.1.0/24 represents an IPv4 service prefix. In a real deployment, the service prefix would be advertised over an EBGP session from an EBGP peer. We have chosen to simply redistribute the prefix locally to simplify this example.
- Family inet is enabled on the IBGP sessions for exchange of IPv4 prefixes between the PE routers.
- On each of the PE nodes, IPv4 service prefix resolves on the LDP route to the BGP next hop of the service prefix.

The output below from R0 shows the OSPF and LDP routes to reach 7.7.7.7, which is the loopback of R7:

```
root@R0> show route 7.7.7.7

inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32        *[OSPF/10] 00:17:17, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0
                     to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32        *[LDP/9] 00:06:24, metric 1
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 299920
                     to 21.0.3.2 via ge-0/0/2.0, Push 299872
```

Figure 1.2 shows all of the interface addresses on the links in the example network. As you can see, the interface subnets are configured following the simple scheme of 21.*X*.*Y*.1(2)/24, where X and Y correspond to two routers that the link connects: RX and RY. The interface addresses are very useful for understanding the next hops in the route table.

For example, in the receding output, the next hop of the LDP route to reach 7.7.7.7 is "21.0.2.2 via ge-0/0/1.0, Push 299920". From the interfaces shown in Figure 1.2, one can readily see that this next hop corresponds to the interface from R0 to R2. So this route corresponds to R0 pushing label 299920 on the packet and sending it out the interface to R2.

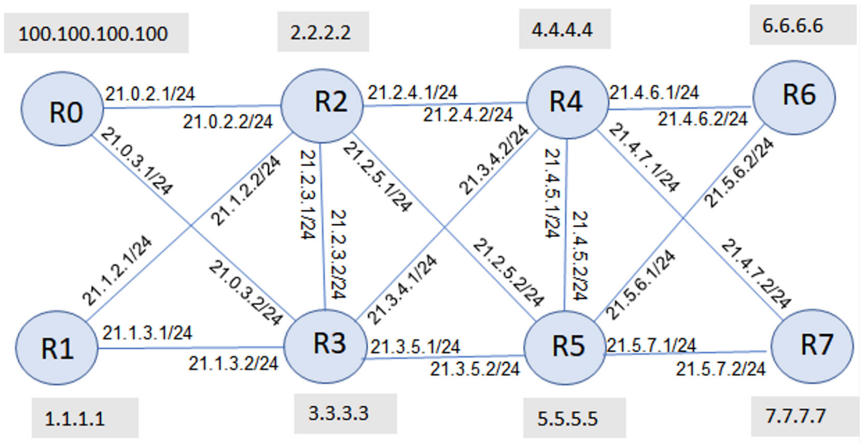


Figure 1.2 Interface Addresses on the Example Network

Now, if we want to learn the route to the external network represented by destination 131.1.1.1 as seen from router R0, enter the show command below and note that the next hop is 21.0.3.2 which is, as we've seen before, the link R0-R3:

```
root@R0> show route 131.1.0/24

inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

131.1.1.0/24       *[BGP/170] 00:18:58, localpref 100, from 7.7.7.7
                      AS path: I, validation-state: unverified
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 299920
                      to 21.0.3.2 via ge-0/0/2.0, Push 299872
```

In the Junos OS, the default behavior is for a BGP next hop to be resolved using the route in inet.0 or inet.3 with the lowest route preference. Since the BGP route 131.1.1.0/24 is being received from 7.7.7.7, and the route for 7.7.7.7 with the lowest preference is the LDP route, the route for 131.1.1.0/24 uses the next hops (the outgoing interface and MPLS labels) provided by the LDP route.

Building Your Own Testbed

We encourage everyone, while reading through this book, to build the topology in their own lab as there is no better way to learn something new than by actually configuring it. Hence all configurations to get you started quickly are attached in Appendix A.

However, like most of us you probably won't have a full rack of equipment available to build your lab on. And running a bunch of virtual routers (vMX) on your laptop could be a challenge as well.

This is where Juniper Networks vLabs (https://jlabs.juniper.net/vlabs/) can help out. vLabs are virtualized "hands-on" labs services that are designed to allow partners, customers, and others interested in learning about the functionality of Juniper's products and solutions. vLabs provides a variety of preset topologies in routing, switching, and security to give you a chance to try out Juniper technologies. vLabs access is free to the public.

The interesting part in this specific case is the SR basic sandbox (https://jlabs.juniper.net/vlabs/portal/sr-basic/index.page), which offers a preconfigured topology consisting of seven vMX routers. As you might have noticed this isn't exactly the topology we are using throughout this book, but it provides a very low cost and easily available option to start exploring some of the concepts discussed!

You will find all the configurations for each chapter in Appendix A. These should quickly help you get up to speed building your lab and offer you a baseline to start testing. As we give you all the configurations per chapter it should be fairly easy to get started and build specific scenarios. Let's get started.

Chapter 2

Migrating from OSPF-LDP to OSPF-SR

As observed earlier, our setup is running OSPF at this point. In this chapter we will enable OSPF-SR so we can start distributing SR information throughout our network.

The OSPF Extensions for segment routing (OSPF-SR) are standardized in RFC8665 (https://datatracker.ietf.org/doc/rfc8665/) and are now widely adopted by most network equipment vendors including Juniper Networks.

Upgrading Junos OS to Support OSPF-SR

Although this chapter only discusses OSPF-SR, which is supported from Junos OS 16.2R1, the next chapters cover technology, for example TI-LFA, that is not supported in this specific version. Hence in order to be able to progress smoothly, we strongly advise you to upgrade your routers to at least Junos OS 20.2R1, as otherwise you will at one point need to perform the upgrade.

TIP See https://apps.juniper.net/feature-explorer/ for the most up-to-date feature and version information.

Enabling Segment Routing in OSPF

To enable OSPF-SR on a single router, apply the following line of configuration:

```
R7# set protocols ospf source-packet-routing node-segment ipv4-index 107
```

When committed (R7# *commit*), this configuration statement causes OSPF to do several different things:

- It causes OSPF to advertise adjacency segments.
- It causes OSPF to advertise an IPv4 node-SID with a value of 107.
- It causes OSPF to advertise a SRGB. We will dive into SRGB later on.
- It causes OSPF to install forwarding entries related to the SR advertisements it sends and receives.

IPv4 OSPF-SR support is enabled through MPLS. OSPF creates a given interface, adjacency, and area per OSPF neighbor. A separate MPLS label is allocated for each adjacency segment created.

Labels are allocated only when the neighbor moves from *init state* to *up state* and requests the label manager for an unreserved label. The corresponding label transitions are downloaded to the MPLS forwarding table after the label is advertised in locally-originated LSAs. In case of LAN (shared Ethernet segment) adjacencies, OSPF neighborship remains in a *two-way state* for the adjacencies between the designated router (DR) and others. A separate label is allocated for each of the LAN neighbors, including the DR-other adjacencies that remain in the two-way state.

The Juniper OSPF implementation enables the network operator to provision the IPv4 address family node segment index *node-SID*. This node-SID will be assigned to a router and used by all other remote routers in the network to index into respective node segment label blocks (SRGBs). It derives the segment identifier to forward IPv4 traffic destined for the same router which was assigned as node-SID.

In the configuration statement above, no SRGB is explicitly mentioned, so Junos dynamically chooses the SRGB. One can see the value of the SRGB that Junos has assigned dynamically by looking at the `show ospf overview` output. However, many network operators choose to explicitly configure the value of the SRGB used in the network, which is what we have chosen to do in this book as well.

Explicitly Configuring the Segment Routing Global Block

The SRGB is the range of MPLS labels used by segment routing. Each IP prefix has an index value associated with it. An SRGB defines a set of MPLS labels. The MPLS label used for forwarding a particular prefix is basically determined by finding the MPLS label in the SRGB corresponding to the index of the prefix.

Explicitly configuring the same SRGB across all of the routers in the SR domain makes the network easier to troubleshoot, since it makes it easier to figure out the meaning of a given SR-MPLS label. Using the same SRGB across the network also has advantages when using anycast-SIDs.

In order to configure an SRGB containing 5000 labels, starting at label 400000, you would use the following configuration in Junos:

```
set protocols ospf source-packet-routing srgb index-range 5000 start-label 400000
```

After committing this configuration, you can see the operational state for OSPF's SRGB by issuing the `show ospf overview` command:

```
regress@R0# run show ospf overview
Instance: master
  Router ID: 100.100.100.100
  Route table index: 0
  LSA refresh time: 50 minutes
  SPRING: Enabled
            SRGB Config Range :
            SRGB Start-Label : 400000, SRGB Index-Range : 5000
            SRGB Block Allocation: Success
            SRGB Start Index : 400000, SRGB Size : 5000, Label-Range: [ 400000, 404999 ]
            Node Segments: Enabled
            Ipv4 Index : 100
  Post Convergence Backup: Disabled
  Area: 0.0.0.0
            Stub type: Not Stub
            Authentication Type: None
            Area border routers: 0, AS boundary routers: 0
            Neighbors
            Up (in full state): 2
  Topology: default (ID 0)
            Prefix export count: 0
            Full SPF runs: 28
            SPF delay: 0.200000 sec, SPF holddown: 5 sec, SPF rapid runs: 3
            Backup SPF: Not Needed
```

SR standards and implementations support different nodes advertising different SRGBs. However, we will keep our examples simple by having all of the nodes in the network advertise the same SRGB for a given IGP. So, in our example, all eight routers are configured to advertise an SRGB starting at label 400000 with a size of 5000.

Figure 2.1 shows the SRGBs and node-SID index values configured on the eight routers in our example topology.

Note that Figure 2.1 refers to the OSPF SRGB. In the OSPF to ISIS migration described later in this book we have OSPF and ISIS use different SRGBs. We have already configured OSPF to use an SRGB of size 5000 with starting label 400000. In Chapter 4 you will configure ISIS to use an SRGB of size 5000 with starting label 200000. This allows you to easily use the same node and prefix-SID index values for both OSPF and ISIS without creating MPLS label conflicts.

Besides configuring different SRGBs, there is another approach to avoiding MPLS label conflicts when running SR during IGP migrations. Junos also supports having OSPF and ISIS advertise and use a single shared SRGB. In Chapter 7 we discuss how to use a shared SRGB along with a few trade-offs between the two approaches. For the moment, let's move forward using different SRGBs for OSPF and ISIS.

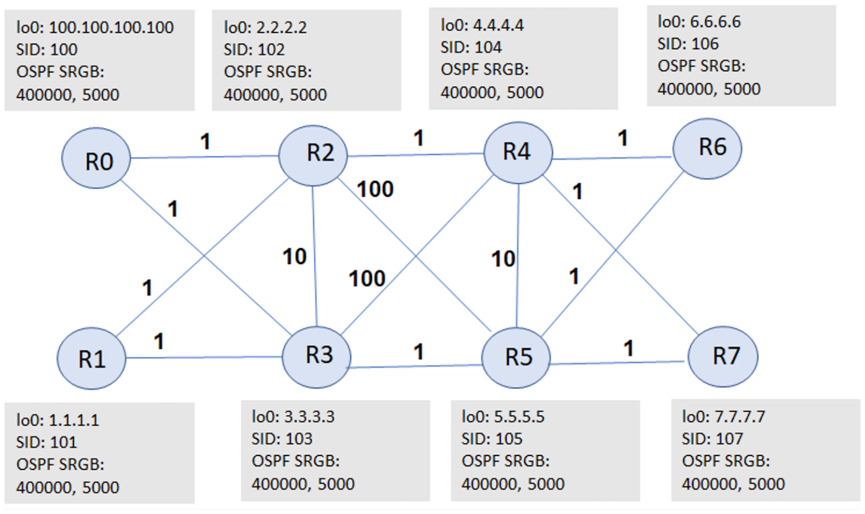


Figure 2.1 *SRGB and Node-SID Configuration for OSPF-LDP to OSPF-SR Migration*

At this point, all of the routers in the network should be advertising a unique node-SID value as well as a common SRGB value. Let's take a quick look at the effect this has had on different routing tables in the network, focusing on the route entries involving 7.7.7.7/32, the loopback address of R7.

On R0, take a look at the route installed by OSPF SR in inet.3 to reach 7.7.7.7. The route is installed by labeled OSPF (L-OSPF) as seen in the output below:

```
regress@R0> show route 7.7.7.7/32 table inet.3 protocol ospf

7.7.7.7/32      [L-OSPF/10/5] 00:02:49, metric 3
               >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                  to 21.0.3.2 via ge-0/0/2.0, Push 400107
```

The route for 7.7.7.7 on R0 has two ECMP next hops. The traffic is load-balanced across the R0-R2 interface and the R0-R3 interface. R0 pushes the label 400107 on the outgoing packets. R0 computes the label value 400107 by adding the prefix-SID index for 7.7.7.7/32 (107) to the common SRGB start label (400000).

Default Preference of LDP Over L-OSPF Routes

At this point, however, note that we still have the LDP route for 7.7.7.7/32 in inet.3 in addition to the L-OSPF route. This can be seen by looking at a less restricted form of the `show route` command:

```
regress@R0> show route 7.7.7.7/32

inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
7.7.7.7/32     *[OSPF/10/10] 01:28:06, metric 3
               >  to 21.0.2.2 via ge-0/0/1.0
                  to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 14 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32     *[LDP/9] 00:10:24, metric 1
               >  to 21.0.2.2 via ge-0/0/1.0, Push 299920
                  to 21.0.3.2 via ge-0/0/2.0, Push 299872
                [L-OSPF/10/5] 2w2d 21:18:00, metric 3
               >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                  to 21.0.3.2 via ge-0/0/2.0, Push 400107
```

In Junos, the default behavior is for a BGP next hop to be resolved using the route in inet.0 or inet.3 with the lowest route preference. LDP routes are installed with a route preference of 9 while internal OSPF and L-OSPF routes are both installed with a route preference of 10.

Let's verify that the IPv4 service route for 131.1.1.0/24 installed by BGP is still using the LDP next hops:

```
regress@R0> show route 131.1.1.0/24

inet.0: 71 destinations, 71 routes (71 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

131.1.1.0/24   *[BGP/170] 00:14:57, localpref 100, from 7.7.7.7
                AS path: I, validation-state: unverified
               >  to 21.0.2.2 via ge-0/0/1.0, Push 299920
                  to 21.0.3.2 via ge-0/0/2.0, Push 299872
```

In Chapter 3 we show how one can modify the route preference for LDP so that L-OSPF routes are preferred over LDP routes. This can be useful in some migration scenarios. However, in this section, let's simply deactivate LDP so that we have only L-OSPF routes to use for forwarding. This will allow us to explore how SR forwarding works without being distracted by the presence of the LDP routes. In Chapter 3, we will turn LDP back on in order to explore more fine-grained approaches to migration.

So let's deactivate LDP on R0 using:

```
deactivate protocols ldp
```

And now let's verify that the LDP route to 7.7.7.7 on R0 has disappeared, leaving only the OSPF route in inet.0 and the L-OSPF route in inet.3:

```
regress@R0>  show route 7.7.7.7

inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32     *[OSPF/10/10] 01:28:06, metric 3
               >  to 21.0.2.2 via ge-0/0/1.0
                  to 21.0.3.2 via ge-0/0/2.0
```

```
inet.3: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32      *[L-OSPF/10/5] 00:05:14, metric 3
                >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                   to 21.0.3.2 via ge-0/0/2.0, Push 400107
```

Note that the OSPF and L-OSPF routes are both installed with a preference value of 10. In the case of a preference tie between OSPF and L-OSPF, BGP route resolution prefers L-OSPF. Let's verify that the BGP route is being resolved over the L-OSPF route:

```
regress@R0>  show route 131.1.1.1

inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

131.1.1.0/24    *[BGP/170] 00:02:45, localpref 100, from 7.7.7.7
                AS path: I, validation-state: unverified
                >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                   to 21.0.3.2 via ge-0/0/2.0, Push 400107
```

Forwarding Along an OSPF-SR Path

IPv4 traffic with DA=131.1.1.1, for example, arriving at R0 will be encapsulated with MPLS label 400107 and sent over ECMP paths to R2 and R3. If you follow the path of this packet to R2, you can see the route for incoming label = 400107 in the mpls.0 table:

```
regress@R2> show route label 400107 table mpls.0

mpls.0: 36 destinations, 36 routes (36 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

400107          *[L-OSPF/10/5] 00:06:55, metric 2
                >  to 21.2.4.2 via ge-0/0/3.0, Swap 400107
```

Traffic arriving at R2 with label 400107 will leave out the R2-R4 interface with label 400107. Since we are using the same SRGB on all nodes in the network, it is easier to track label values across different routers. As can be seen in the next output, traffic arriving at R4 with label 400107 will have that label popped and the packet will be sent out of interface R4-R7 with the original IP packet exposed:

```
regress@R4> show route label 400107 table mpls.0

mpls.0: 36 destinations, 36 routes (36 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

400107          *[L-OSPF/10/5] 00:09:26, metric 1
                >  to 21.4.7.2 via ge-0/0/6.0, Pop
400107(S=0)     *[L-OSPF/10/5] 00:09:26, metric 1
                >  to 21.4.7.2 via ge-0/0/6.0, Pop
```

Enabling TI-LFA

In its simplest form, TI-LFA provides protection against the failure of a single link. The TI-LFA backup path for a particular destination corresponds to the path that traffic will eventually take once the IGP converges after the failure of a link. This is referred to as the post-convergence backup path. In order to make traffic follow the post-convergence path, TI-LFA can use several labels in the label stack that define the backup path.

The Junos TI-LFA implementation also supports protecting against the possibility of multiple correlated link failures. These can be configured using the node-protection, fate-sharing-protection, and SRLG-protection options. However, in this example let's configure TI-LFA to protect against the failure of a single link.

Enable TI-LFA for OSPF on the two interfaces on R0 using the following:

```
set protocols ospf backup-spf-options use-post-convergence-lfa
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 post-convergence-lfa
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

The first line is needed to activate TI-LFA for OSPF in general and also serves as a place to configure several options related to TI-LFA. The second and third lines enable the computation and installation of the link-protecting post-convergence path on the two interfaces on R0.

The last line changes the maximum-backup-paths parameter used by TI-LFA. By default, the Junos TI-LFA implementation will only install one backup path, even when several equal-cost post-convergence backup paths exist. By setting maximum-backup-paths equal to 8, Junos will install up to eight equal-cost post-convergence backup paths.

Now go ahead and enable TI-LFA on all interfaces on all routers in the testbed. You can consult the configuration in Appendix A for details.

Applying this configuration will not directly affect the way that traffic flows. It will only install backup paths that will be used for a short period of time in the event of failures.

OSPF-SR inet.3 Routes

When OSPF-SR is enabled together with TI-LFA on all of the routers using the basic configurations described here, a labeled OSPF route for each loopback prefix is created and downloaded to the inet.3 and the mpls.0 route tables. All of the labeled OSPF routes in inet.3 correspond to node-SID advertisements. Here are the labeled OSPF routes that one expects to see in the inet.3 table on R0:

```
regress@R0> show route table inet.3

inet.3: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1/32      *[L-OSPF/10/5] 18:03:43, metric 2
                >  to 21.0.2.2 via ge-0/0/1.0, Push 400101
                   to 21.0.3.2 via ge-0/0/2.0, Push 400101
2.2.2.2/32      *[L-OSPF/10/5] 18:03:38, metric 1
                >  to 21.0.2.2 via ge-0/0/1.0
                   to 21.0.3.2 via ge-0/0/2.0, Push 400102, Push 400101(top)
3.3.3.3/32      *[L-OSPF/10/5] 18:03:38, metric 1
                >  to 21.0.3.2 via ge-0/0/2.0
                   to 21.0.2.2 via ge-0/0/1.0, Push 400103, Push 400101(top)
4.4.4.4/32      *[L-OSPF/10/5] 16:41:41, metric 2
                >  to 21.0.2.2 via ge-0/0/1.0, Push 400104
                   to 21.0.3.2 via ge-0/0/2.0, Push 400104, Push 400101(top)
                   to 21.0.3.2 via ge-0/0/2.0, Push 400104, Push 400106(top)
                   to 21.0.3.2 via ge-0/0/2.0, Push 400104, Push 400107(top)
5.5.5.5/32      *[L-OSPF/10/5] 16:41:41, metric 2
                >  to 21.0.3.2 via ge-0/0/2.0, Push 400105
                   to 21.0.2.2 via ge-0/0/1.0, Push 400105, Push 400101(top)
                   to 21.0.2.2 via ge-0/0/1.0, Push 400105, Push 400106(top)
                   to 21.0.2.2 via ge-0/0/1.0, Push 400105, Push 400107(top)
6.6.6.6/32      *[L-OSPF/10/5] 18:03:13, metric 3
                >  to 21.0.2.2 via ge-0/0/1.0, Push 400106
                   to 21.0.3.2 via ge-0/0/2.0, Push 400106
7.7.7.7/32      *[L-OSPF/10/5] 18:03:03, metric 3
                >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                   to 21.0.3.2 via ge-0/0/2.0, Push 400107
```

Inet.3 Route to R7

If you look at the network topology in Figure 2.1, you can make sense of this output. For example, there are two shortest paths from R0 to R7: R0->R2->R4->R7 and R0->R3->R5->R7; both have a cost of 3. The two next hops shown for 7.7.7.7/32 correspond to those two shortest paths. When there are multiple equal-cost shortest paths, the TI-LFA implementation does not install any additional backup paths (except in some cases involving node-protection or SRLG-protection.) One can verify that traffic will be load-balanced across the two next hops to reach 7.7.7.7/32 by looking at the value of the `weight` in the output of `show route detail`:

```
regress@R0> show route 7.7.7.7 table inet.3 detail | match weight
                Next hop: 21.0.2.2 via ge-0/0/1.0 weight 0x1, selected
                Next hop: 21.0.3.2 via ge-0/0/2.0 weight 0x1
```

Since both next hops have the same weight value of 0x1, the traffic will be load-balanced across interfaces ge-0/0/1.0 and ge-0/0/2.0. If one of those interfaces goes down, all of the traffic will automatically be sent on to the remaining interface. Therefore, no additional backup path is needed.

Inet.3 Route to R2

Look at the primary and backup paths to get from R0 to R2. The shortest path from R0 to R2 is simply across the link R0->R2 with cost of 1. If the link from R0 to R2 were to fail, the shortest path from R0 to R2 would be along the path R0->R3->R1->R2. The route for 2.2.2.2/32 can then be understood in this context:

```
regress@R0> show route 2.2.2.2 table inet.3

inet.3: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2.2.2.2/32     *[L-OSPF/10/5] 18:47:07, metric 1
               >  to 21.0.2.2 via ge-0/0/1.0
                  to 21.0.3.2 via ge-0/0/2.0, Push 400102, Push 400101(top)
```

This first next hop corresponds to the primary path from R0->R2, going out ge-0/0/1 directly to R2. The second next hop corresponds to the backup path R0->R3->R1->R2. The packet leaves R0 out interface ge-0/0/2.0 to R3 with a label stack [400101(top),400102]. R3 will send the packet out its interface to R1 with label stack [400102(top)]. And R1 will send the packet out its interface to R2 with no label stack.

Let's verify that R0 will only use the backup path when the R0-R2 interface fails. This can be done by looking at the value of the `weight` in the output of a `show route detail` command:

```
egress@R0> show route 2.2.2.2 table inet.3 detail | match weight
           Next hop: 21.0.2.2 via ge-0/0/1.0 weight 0x1, selected
           Next hop: 21.0.3.2 via ge-0/0/2.0 weight 0xf000
```

The next hop to R3 via ge-0/0/2.0 has `weight 0xf000`, so it will only be used if the next hop to R2 via ge-0/0/1.0 becomes unusable.

Inet.3 Route to R4

Next look at the primary and backup paths to get from R0 to R4. The shortest path from R0 to R4 follows R0->R2->R4 with a cost of 2. This accounts for the first next hop in the `show route` output:

```
regress@R0> show route 4.4.4.4 table inet.3

inet.3: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

4.4.4.4/32     *[L-OSPF/10/5] 22:14:22, metric 2
               >  to 21.0.2.2 via ge-0/0/1.0, Push 400104
                  to 21.0.3.2 via ge-0/0/2.0, Push 400104, Push 400101(top)
                  to 21.0.3.2 via ge-0/0/2.0, Push 400104, Push 400106(top)
                  to 21.0.3.2 via ge-0/0/2.0, Push 400104, Push 400107(top)
```

The other three next hops in the show route output correspond to the three different equal-cost backup paths to reach R4 installed by the TI-LFA. If the link from R0 to R2 were to fail, the three different shortest paths to reach R4 would be R0->R3->R1->R2->R4, R0->R3->R5->R6->R4, and R0->R3->R5->R7->R4, each with cost of 4.

It is a worthwhile exercise to study the network topology in Figure 2.1, and prove to yourself that the last three outgoing interfaces and label stacks in the show route output do in fact correspond to post-convergence shortest paths to reach R4 when the interface to R2 fails.

BGP Route Inherits TI-LFA Backup Next hops

With the existing topology, the BGP route on R0 for 131.1.1.0/24 (advertised by R7) does not have a TI-LFA backup path, because the primary path has ECMP. It's useful to temporarily modify the topology in Figure 2.1 to illustrate how BGP routes resolving on OSPF-SR route inherit the TI-LFA backup next hops. To do this deactivate the interface on R3 to R5:

```
regress@R3# deactivate interfaces ge-0/0/5.0
```

With the link from R3 to R5 deactivated, the primary path for R0 to reach R7 is along R0->R2->R4->R7, while the TI-LFA backup path is R0->R3->R1->R2->R4->R7. This is now reflected in the inet.3 route to reach 7.7.7.7/32:

```
regress@R0> show route 7.7.7.7 table inet.3

inet.3: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32      *[L-OSPF/6/5] 00:05:58, metric 3
                >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                   to 21.0.3.2 via ge-0/0/2.0, Push 400107, Push 400101(top)
```

Importantly, the BGP route for 131.1.1.0/24 has inherited the TI-LFA backup path to reach R7:

```
regress@R0> show route 131.1.1.0/24

inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

131.1.1.0/24    *[BGP/170] 15:51:55, localpref 100, from 7.7.7.7
                AS path: I, validation-state: unverified
                >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                   to 21.0.3.2 via ge-0/0/2.0, Push 400107, Push 400101(top)
```

So the routes that get the service traffic into MPLS tunnels benefit from the hardware level fast-reroute provided by the TI-LFA backup paths.

Please remember to return the example topology to its original state by activating the link on R3 to R5 using:

```
regress@R3# activate interfaces ge-0/0/5.0
```

MORE? For more details on the implementation of TI-LFA in Junos and other TI-LFA configuration options, make sure to read *Day One: Configuring Segment Routing with Junos* at https://www.juniper.net/documentation/en_US/day-one-books/DO_InsideSR.zip.

OSPF-SR mpls.0 Routes

Now let's look at the OSPF-SR routes that get installed in the mpls.0 table on R0. Labeled OSPF produces two types of entries in the mpls.0 table, corresponding to adjacency-SIDs and node-SIDs. The `show route output` produces:

```
regress@R0> show route table mpls.0 protocol ospf

mpls.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

16            *[L-OSPF/10/5] 18:09:34, metric 0
                >  to 21.0.2.2 via ge-0/0/1.0, Pop
                   to 21.0.3.2 via ge-0/0/2.0, Swap 400102, Push 400101(top)
16(S=0)         *[L-OSPF/10/5] 18:09:34, metric 0
                >  to 21.0.2.2 via ge-0/0/1.0, Pop
                   to 21.0.3.2 via ge-0/0/2.0, Swap 400102, Push 400101(top)
17            *[L-OSPF/10/5] 18:09:34, metric 0
                >  to 21.0.3.2 via ge-0/0/2.0, Pop
                   to 21.0.2.2 via ge-0/0/1.0, Swap 400103, Push 400101(top)
17(S=0)         *[L-OSPF/10/5] 18:09:34, metric 0
                >  to 21.0.3.2 via ge-0/0/2.0, Pop
                   to 21.0.2.2 via ge-0/0/1.0, Swap 400103, Push 400101(top)
400101          *[L-OSPF/10/5] 18:09:39, metric 2
                   to 21.0.2.2 via ge-0/0/1.0, Swap 400101
                >  to 21.0.3.2 via ge-0/0/2.0, Swap 400101
400102          *[L-OSPF/10/5] 18:09:34, metric 1
                >  to 21.0.2.2 via ge-0/0/1.0, Pop
                   to 21.0.3.2 via ge-0/0/2.0, Swap 400102, Push 400101(top)
400102(S=0)     *[L-OSPF/10/5] 18:09:34, metric 1
                >  to 21.0.2.2 via ge-0/0/1.0, Pop
                   to 21.0.3.2 via ge-0/0/2.0, Swap 400102, Push 400101(top)
400103          *[L-OSPF/10/5] 18:09:34, metric 1
                >  to 21.0.3.2 via ge-0/0/2.0, Pop
                   to 21.0.2.2 via ge-0/0/1.0, Swap 400103, Push 400101(top)
400103(S=0)     *[L-OSPF/10/5] 18:09:34, metric 1
                >  to 21.0.3.2 via ge-0/0/2.0, Pop
                   to 21.0.2.2 via ge-0/0/1.0, Swap 400103, Push 400101(top)
400104          *[L-OSPF/10/5] 16:47:37, metric 2
                >  to 21.0.2.2 via ge-0/0/1.0, Swap 400104
                   to 21.0.3.2 via ge-0/0/2.0, Swap 400104, Push 400101(top)
                   to 21.0.3.2 via ge-0/0/2.0, Swap 400104, Push 400106(top)
                   to 21.0.3.2 via ge-0/0/2.0, Swap 400104, Push 400107(top)
```

```
400105         *[L-OSPF/10/5] 16:47:37, metric 2
               >  to 21.0.3.2 via ge-0/0/2.0, Swap 400105
                  to 21.0.2.2 via ge-0/0/1.0, Swap 400105, Push 400101(top)
                  to 21.0.2.2 via ge-0/0/1.0, Swap 400105, Push 400106(top)
                  to 21.0.2.2 via ge-0/0/1.0, Swap 400105, Push 400107(top)
400106         *[L-OSPF/10/5] 18:09:09, metric 3
                  to 21.0.2.2 via ge-0/0/1.0, Swap 400106
               >  to 21.0.3.2 via ge-0/0/2.0, Swap 400106
400107         *[L-OSPF/10/5] 18:08:59, metric 3
                  to 21.0.2.2 via ge-0/0/1.0, Swap 400107
               >  to 21.0.3.2 via ge-0/0/2.0, Swap 400107
```

Adjacency-SID Routes in mpls.0

The entries for incoming label 16 and incoming label 17 correspond to the two adjacency-SIDs advertised by R0. The label values of the adjacency-SIDs have been dynamically allocated, so one may see different label values. You can confirm that these are the label values that were dynamically assigned by using `show ospf neighbor detail`:

```
regress@R0> show ospf neighbor detail
Address      Interface        State      ID           Pri  Dead
21.0.2.2     ge-0/0/1.0         Full       2.2.2.2      128  30
  Area 0.0.0.0, opt 0x52, DR 0.0.0.0, BDR 0.0.0.0
  Up 1d 02:40:15, adjacent 1d 02:40:15
  SPRING Adjacency Labels:

          Label    Flags    Adj-Sid-Type

          16         BVL      Default Dynamic

21.0.3.2     ge-0/0/2.0         Full       3.3.3.3      128  36
  Area 0.0.0.0, opt 0x52, DR 0.0.0.0, BDR 0.0.0.0
  Up 1d 02:40:05, adjacent 1d 02:40:05
  SPRING Adjacency Labels:

          Label    Flags    Adj-Sid-Type

          17         BVL      Default Dynamic
```

Next focus on the mpls.0 route for incoming label 16:

```
regress@R0> show route label 16

mpls.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

16           *[L-OSPF/10/5] 1d 17:19:15, metric 0
               >  to 21.0.2.2 via ge-0/0/1.0, Pop
                  to 21.0.3.2 via ge-0/0/2.0, Swap 400102, Push 400101(top)
16(S=0)        *[L-OSPF/10/5] 1d 17:19:15, metric 0
               >  to 21.0.2.2 via ge-0/0/1.0, Pop
                  to 21.0.3.2 via ge-0/0/2.0, Swap 400102, Push 400101(top)
```

Note that the output shows two entries: 16 and 16(S=0). This allows for different actions based on the setting of the bottom-of-stack bit (S-bit) in the MPLS header. These two entries will generally be the same for adjacency-SID entries in the mpls.0 table, so let's focus on the entry for 16.

The entry for label 16 has two next hops, corresponding to primary and backup paths. The forwarding action for the primary path is to pop the incoming label and send the packet out the interface to R2. The forwarding action for the backup path will result in the packet being sent along the path R0->R3->R1->R2 in order to get to R2. This backup path for the adjacency-SID forwarding entry is installed by the TI-LFA.

Node-SID Routes in mpls.0

The other labeled OSPF entries in the mpls.0 table (for labels 400101-400107) correspond to node-SIDs for R1-R7. You can see that these mpls.0 routes are very similar to the inet.3 routes to reach R1-R7, the main difference being that the mpls.0 routes involve an initial label swap operation to replace the incoming label, where the inet.3 routes involve an initial label push operation.

Verifying OSPF-SR Forwarding Paths with MPLS Ping and Traceroute

Forwarding along the OSPF segment-routing paths can be verified with a MPLS ping and traceroute. You can use MPLS ping to verify the forwarding to reach R7 using the OSPF-SR routes:

```
regress@R0> ping mpls segment-routing ospf 7.7.7.7 detail count 1
Request for seq 1, to interface 357, label 400107, packet size 80
Reply for seq 1, return code: Egress-ok, time: 2.071 ms
   Local transmit time: 2021-01-31 10:39:08 PST 22.809 ms
   Remote receive time: 2021-01-31 10:39:08 PST 24.880 ms

--- lsping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
```

You can use MPLS traceroute to verify the actual paths for forwarding to reach R7 using the OSPF-SR routes. There are two ECMP primary paths from R0 to R7: R0->R2->R4->R7 and R0->R3->R5->R7. The MPLS traceroute command verifies both:

```
regress@R0> traceroute mpls segment-routing ospf 7.7.7.7
  Probe options: ttl 64, retries 3, wait 10, paths 16, exp 7, fanout 16

  ttl      Label  Protocol Address      Previous Hop Probe Status
           1   400107  OSPF     21.0.2.2     (null)       Success
  FEC-Stack-Sent: OSPF
  ttl      Label  Protocol Address      Previous Hop Probe Status
           2  400107  OSPF    21.2.4.2     21.0.2.2     Success
  FEC-Stack-Sent: OSPF
  ttl      Label  Protocol Address      Previous Hop Probe Status
           3    3  OSPF    21.4.7.2     21.2.4.2     Egress
  FEC-Stack-Sent: OSPF

  Path 1 via ge-0/0/1.0 destination 127.0.0.64

  ttl      Label  Protocol Address      Previous Hop Probe Status
           1  400107  OSPF    21.0.3.2     (null)       Success
  FEC-Stack-Sent: OSPF
  ttl      Label  Protocol Address      Previous Hop Probe Status
           2  400107  OSPF    21.3.5.2     21.0.3.2     Success
  FEC-Stack-Sent: OSPF
  ttl      Label  Protocol Address      Previous Hop Probe Status
           3    3  OSPF    21.5.7.2     21.3.5.2     Egress
  FEC-Stack-Sent: OSPF

  Path 2 via ge-0/0/2.0 destination 127.0.1.64
```

You can also run MPLS ping and traceroute commands with an explicit label stack. Note that this requires the following additional configuration on the router originating the MPLS ping or traceroute:

```
regress@R0# set protocols source-packet-routing
```

You can use the MPLS traceroute with an explicit label stack to validate the forwarding path for a TI-LFA backup path. For example, you previously saw that the next hop corresponding to the TI-LFA backup path for the labeled OSPF route to reach 2.2.2.2/32 is:

```
to 21.0.3.2 via ge-0/0/2.0, Push 400102, Push 400101(top)
```

This information can be used to create the following MPLS traceroute command:

```
regress@R0>traceroute mpls segment-routing spring-te label-stack labels [400102 400101] nexthop-
interface ge-0/0/2.0 nexthop-address 21.0.3.2
Probe options: retries 3, exp 7

  ttl      Label  Protocol Address      Previous Hop Probe Status
           1  400101  Static  21.0.3.2     (null)       Success
  FEC-Stack-Sent: SPRING-TE
  ttl      Label  Protocol Address      Previous Hop Probe Status
           1    3  Static  21.1.3.1     21.0.3.2     Success
  FEC-Stack-Sent: SPRING-TE
  ttl      Label  Protocol Address      Previous Hop Probe Status
           2    3  Static  21.1.2.2     21.1.3.1     Egress
  FEC-Stack-Sent: SPRING-TE

  Path 1 via ge-0/0/2.0 destination 127.0.0.64
```

Note that MPLS traceroute using an explicit label-stack does not currently support verifying all ECMP paths. When an incoming label entry has multiple equal-cost next hops, only one of the next hops in the forwarding path will be explored. Support for ECMP paths is being added in future releases.

Verification of the OSPF-SR forwarding entries can be done with or without the LDP routes being present. The MPLS ping and traceroute commands that use IP addresses require a protocol to be specified, so that the correct inet.3 entry can be used independent of which route is active. As discussed earlier, the mpls.0 routes installed by LDP and labeled OSPF will use different incoming labels, so there will be no conflict between mpls.0 routes. Therefore OSPF-SR forwarding paths can be verified before moving the service traffic from LDP paths onto the newly created OSPF-SR paths.

Chapter 3

Shifting Traffic from OSPF-LDP to OSPF-SR

Now let's look at different methods of migrating the actual service traffic from OSPF-LDP to OSPF-SR. The principles introduced in this chapter are initially discussed in the context of shifting traffic from OSPF-LDP to OSPF-SR. However, the same principles will be applied to the other service migration scenarios discussed in this book, such as OSPF-SR to ISIS-SR, and ISIS-LDP to ISIS-SR.

In the previous section we deactivated LDP in order to look at OSPF-SR forwarding behavior without the presence of LDP routes. Now we want to activate LDP again so that we can explore more fine-grained approaches to migration.

Let's go ahead and activate LDP on R0 using:

```
activate protocols ldp
```

After committing, the route table should again have three routes to reach loopback 7.7.7.7 on R7: OSPF in inet.0, L-OSPF in inet.3, and LDP in inet.3:

```
regress@R0> show route 7.7.7.7

inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32       *[OSPF/10/10] 01:28:06, metric 3
                 >  to 21.0.2.2 via ge-0/0/1.0
                    to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 14 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32       *[LDP/9] 01:28:04, metric 1
                 >  to 21.0.2.2 via ge-0/0/1.0, Push 34
                    to 21.0.3.2 via ge-0/0/2.0, Push 30
                  [L-OSPF/10/5] 01:28:06, metric 3
                 >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                    to 21.0.3.2 via ge-0/0/2.0, Push 400107
```

With the default route preference values for OSPF—labeled OSPF, and LDP—we again expect to see the service prefix 131.1.1.0/24 (which is advertised in BGP from 7.7.7.7) being resolved using the LDP route:

```
regress@R0> show route 131.1.1.0/24

inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

131.1.1.0/24      *[BGP/170] 06:17:53, localpref 100, from 7.7.7.7
                    AS path: I, validation-state: unverified
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 34
                    to 21.0.3.2 via ge-0/0/2.0, Push 30
```

The next two sections look at two different methods for controlling how service traffic uses LDP routes or labeled OSPF routes:

- One method is to change the route preference values for LDP and labeled OSPF routes.
- The second method is to use different loopback addresses as the BGP next hops for the service routes and selectively advertise a given loopback into either LDP or OSPF-SR.

Shifting Service Traffic from LDP to OSPF-SR Using Route Preference

Different routing protocols install routes with different route preference values. The route preference value is used to determine the active route when more than one protocol installs the same route (including prefix length). The route with the lowest numeric value of the route preference is preferred and is considered the active route.

Junos allows you to configure the route preference value used by a given protocol. We will set the route preference for L-OSPF routes to be 6, so that L-OSPF routes are preferred over LDP routes (which have a default route preference of 9.)

WARNING Changing route preferences is a very powerful tool, but it should be used carefully. In this example, we reduce the route preference for L-OSPF routes to 6 and only a few protocols are installing routes. A real deployment may have several other protocols installing routes with different preferences. In a real deployment, one needs to take into account all the potential sources of routes when changing route preference values and do thorough testing to avoid surprises.

The Junos implementation has separate configurable values for the OSPF route preferences: `set protocols ospf preference` determines the preference value for inet.0 routes installed by OSPF and `set protocols ospf labeled-preference` determines the preference value for inet.3 routes installed by labeled OSPF (indicated with L-OSPF).

So now, without further ado, let's set the route preference for labeled OSPF routes to be 6, using the following on R0:

```
set protocols ospf labeled-preference 6
```

It has had the desired effect of making the labeled OSPF route the active route as shown here:

```
regress@R0> show route 7.7.7.7

inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32         *[OSPF/10/10] 01:28:06, metric 3
                   >  to 21.0.2.2 via ge-0/0/1.0
                     to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 14 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32         *[L-OSPF/6/5] 00:00:19, metric 3
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                     to 21.0.3.2 via ge-0/0/2.0, Push 400107
                   [LDP/9] 08:04:12, metric 1
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 34
                     to 21.0.3.2 via ge-0/0/2.0, Push 30
```

And you can see next that the service prefix 131.1.1.0/24 is now being resolved over the labeled OSPF route:

```
regress@R0> show route 131.1.1.0/24

inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

131.1.1.0/24       *[BGP/170] 00:07:44, localpref 100, from 7.7.7.7
                     AS path: I, validation-state: unverified
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                     to 21.0.3.2 via ge-0/0/2.0, Push 400107
```

Note that using route preference to migrate from LDP to SR allows for a very gradual and controlled migration. A network operator can change the `ospf labeled-preference` on subsets of nodes in the network over the course of a week in order to look for unexpected behavior in a controlled manner.

In Chapter 6 you will see that the technique of modifying route preferences can also be used to support SR deployments with legacy LDP nodes.

Now let's look at that second alternative to changing route preference, so if you're following along in a similar lab, please remove the changes to R0 that have been made in this section using:

```
delete protocols ospf labeled-preference
```

Shifting Service Traffic from LDP to OSPF-SR using Different Loopback Addresses

One can control how different services use different MPLS tunneling technologies by configuring different loopbacks addresses at the tunnel destination node. This has been a common technique for carrying some services via LDP tunnels and other services via RSVP tunnels. It can be accomplished by advertising two different loopback addresses using two different MPLS protocols, and advertising the BGP service prefixes using the two different loopback addresses as the BGP next hops.

The same technique can be applied with LDP and OSPF-SR. To use this technique, you first need to understand how to advertise a particular loopback address into OSPF-SR with a particular node-SID.

Simple Node-SID Configuration with a Single Loopback Address

Up until this point, we have used the simplest configuration in Junos for advertising a node-SID. In our starting configuration for R7, 7.7.7.7 is configured as the router-id, and 7.7.7.7/32 is the IP address of the interface lo0.0. Interface lo0.0 is configured under protocols OSPF and protocols LDP. This results in a normal (non-SR) OSPF prefix advertisement for 7.7.7.7/32. The simple LDP configuration results in a FEC for 7.7.7.7/32 being advertised, since it corresponds to the router-id.

In Chapter 2's *Enabling SR in OSPF*, we applied the following additional configuration to R7:

```
R7# set protocols ospf source-packet-routing node-segment ipv4-index 107
```

This configuration caused R7 to advertise that the index value 107 is associated with 7.7.7.7/32, since this is the prefix corresponding to the router-id on R7.

With this simple node-SID configuration, both LDP and OSPF-SR create MPLS paths to reach 7.7.7.7. As shown in Chapter 2's *Default Preference of LDP Over L-OSPF Routes*, the active route in inet.3 for 7.7.7.7/32 will determine how the route for BGP service prefix with next-hop 7.7.7.7 is resolved.

Policy-Based Node-SID Configuration with Multiple Loopback Addresses

Let's configure a second loopback address on R7 (77.77.77.77/32) and advertise it into OSPF together with a node-SID advertisement (with index value 507):

```
set interfaces lo0.0 family inet address 77.77.77.77/32

set policy-options policy-statement assign_sid term one from protocol direct
set policy-options policy-statement assign_sid term one from route-filter 77.77.77.77/32 exact
set policy-options policy-statement assign_sid term one then prefix-segment index 507
set policy-options policy-statement assign_sid term one then prefix-segment node-segment
set policy-options policy-statement assign_sid term one then accept

set protocols ospf source-packet-routing prefix-segment assign_sid
```

The configuration above uses the familiar Junos policy configuration with new `prefix-segment` actions, together with the `prefix-segment` configuration under `protocols ospf source-packet-routing`.

Once the above configuration is committed, you can see next that R0 has a labeled OSPF route in inet.3 for 77.77.77.77/32:

```
regress@R0> show route 77.77.77.77 table inet.3

inet.3: 8 destinations, 15 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

77.77.77.77/32   *[L-OSPF/10/5] 00:02:10, metric 3
                 >  to 21.0.2.2 via ge-0/0/1.0, Push 400507
                    to 21.0.3.2 via ge-0/0/2.0, Push 400507, Push 400101(top)
```

Note that this configuration on R7 does not result in 77.77.77.77 being advertised into LDP, so the only route on R0 for 77.77.77.77 will correspond to the labeled OSPF route.

R0 has both LDP and L-OSPF routes for 7.7.7.7 but with the current configuration the LDP route is preferred:

```
regress@R0> show route 7.7.7.7 table inet.3

inet.3: 8 destinations, 15 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32       *[LDP/9] 18:02:36, metric 1
                 >  to 21.0.2.2 via ge-0/0/1.0, Push 58
                 [L-OSPF/10/5] 18:02:32, metric 3
                 >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                    to 21.0.3.2 via ge-0/0/2.0, Push 400107, Push 400101(top)
```

The final step is to make a particular BGP service route advertised by R7 use 77.77.77.77 as its BGP next hop:

```
set policy-options policy-statement bgp_export_policy term one from route-
filter 131.1.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term one then next-hop 77.77.77.77
set policy-options policy-statement bgp_export_policy term one then accept
```

Now a network operator can control which service traffic will use the LDP MPLS tunnel and which will use the OSPF-SR MPLS tunnel to reach R7, based on the BGP next hop advertised for the service. This can be used as part of a migration strategy, or longer term if desired.

Setting the Node Flag In a Prefix-SID Advertisement

Notice that the configuration on R7 in the preceding section has the following line:

```
set policy-options policy-statement assign_sid term one then prefix-segment node-segment
```

This results in the node segment flag being set in the prefix-SID advertisement for 77.77.77.77/32. When a prefix-SID advertisement has the node flag set, we usually refer to it as a *node-SID*. One should set the node flag for a prefix advertisement when the prefix is expected to only be advertised by one node. The loopback address 77.77.77.77 should only be advertised by R7, so it meets the requirement for setting the node flag.

The fact that a particular prefix-SID should only be advertised by one node in the network is useful when constructing paths using prefix-SIDs. TI-LFA can make use of information to simplify computations for constructing post-convergence backup paths.

Chapter 4

Preparing for a Migration From OSPF-SR to ISIS-SR

Now let's look at migrating from OSPF-SR to ISIS-SR. For this chapter we start with the network running only OSPF-SR, add ISIS-SR running in parallel, and then shift traffic from OSPF-SR to ISIS-SR. In Chapter 6 we'll discuss migration from ISIS-LDP to ISIS-SR. For the current chapter, let's go ahead and deactivate LDP on all of the routers by using:

```
deactivate protocols ldp
```

ISIS-SR Configuration

You want to enable ISIS-SR on all of the nodes. The configurations for all of the routers can be found in Appendix A.

To quickly review the configuration done on R7:

```
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/5.0 level 1 disable
set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface ge-0/0/6.0 level 1 disable
set protocols isis interface ge-0/0/6.0 level 2 metric 1
set protocols isis interface lo0.0 passive
```

You can see a basic Junos OS configuration for level 2 ISIS, without any segment routing. In this migration example, we want to use the same metrics in ISIS as in OSPF. Note that you need to configure metric 1 on the ISIS interface to match the default metric on OSPF interfaces:

```
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 107

set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis interface ge-0/0/5.0 level 2 post-convergence-lfa
set protocols isis interface ge-0/0/6.0 level 2 post-convergence-lfa
```

The first two lines of the configuration create an SRGB of size 5000, starting with label 200000, and advertise that SRGB. The third line generates a node-SID advertisement with index 107 for the loopback address 7.7.7.7/32. The last three lines enable the installation of TI-LFA backup paths.

In this example we configured ISIS and OSPF to use different SRGBs. ISIS is using an SRGB of size 5000 and starting label 200000. OSPF is using an SRGB of the same size and starting at label 400000. This allows us to use the same index values for the same prefixes, without creating MPLS label conflicts.

The result of the complete configuration is shown in Figure 4.1.

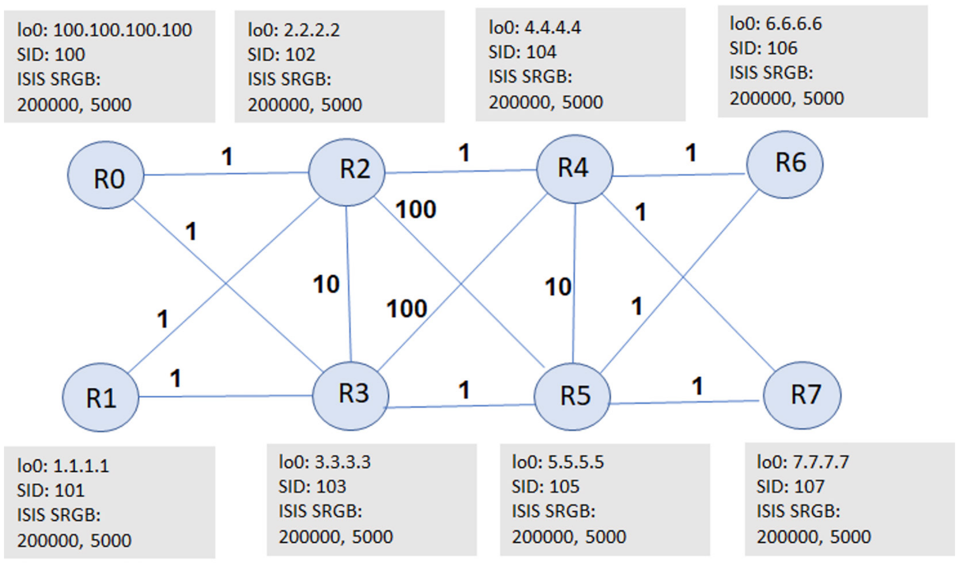


Figure 4.1 SRGB and Node-SID Configuration for OSPF-SR to ISIS-SR Migration

ISIS-SR Node-SID Routes

On R0, you can see next that both ISIS-SR and OSPF-SR contribute routes to the inet.3 table to reach 7.7.7.7/32. The L-OSPF route uses label 400107, while the L-ISIS route uses label 200107:

```
regress@R0> show route 7.7.7.7 table inet.3

inet.3: 7 destinations, 14 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32         *[L-OSPF/10/5] 00:16:05, metric 3
                    >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                      to 21.0.3.2 via ge-0/0/2.0, Push 400107
                    [L-ISIS/14] 00:15:13, metric 3
                    >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                      to 21.0.3.2 via ge-0/0/2.0, Push 200107
```

The mpls.0 entries for labels 200107 and 400107 on R2 show how the ISIS-SR and OSPF-SR MPLS forwarding planes coexist, each using its own SRGB for node and prefix-SIDs:

```
root@R2# run show route label 200107

mpls.0: 67 destinations, 67 routes (67 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

200107             *[L-ISIS/14] 00:00:04, metric 2
                    >  to 21.2.4.2 via ge-0/0/3.0, Swap 200107
                      to 21.0.2.1 via ge-0/0/0.0, Swap 200107, Push 200105(top)
                      to 21.1.2.1 via ge-0/0/1.0, Swap 200107, Push 200105(top)

regress@R2> show route label 400107

mpls.0: 67 destinations, 67 routes (67 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

400107             *[L-OSPF/10/5] 21:31:53, metric 2
                    >  to 21.2.4.2 via ge-0/0/3.0, Swap 400107
                      to 21.0.2.1 via ge-0/0/0.0, Swap 400107, Push 400105(top)
                      to 21.1.2.1 via ge-0/0/1.0, Swap 400107, Push 400105(top)
```

ISIS-SR Adjacency-SID Routes

You can also verify the entries corresponding to adjacency SID in the mpls.0 table on R2:

```
root@R2# run show isis adjacency R4 detail
R4
  Interface: ge-0/0/3.0, Level: 2, State: Up, Expires in 26 secs
  Priority: 0, Up/Down transitions: 3, Last transition: 00:00:38 ago
  Circuit type: 2, Speaks: IP, IPv6
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  IP addresses: 21.2.4.2
  Level 2 IPv4 Adj-SID: 36
```

The output from R2 above shows that ISIS has assigned label value 36 for the adjacency-SID on ge-0/0/3.0 to reach R4.

If you look at the output from that route entry on R2 for incoming label 36, you'll see that the primary forwarding action corresponds to popping the label and forwarding out ge-0/0/3.0. The backup forwarding action (in case ge-0/0/3.0 fails) gets the packet to R4 via ECMP post-convergence backup paths:

```
root@R2# run show route label 36

mpls.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

36                 *[L-ISIS/14] 02:43:14, metric 0
                   >  to 21.2.4.2 via ge-0/0/3.0, Pop
                      to 21.0.2.1 via ge-0/0/0.0, Swap 200104, Push 200105(top)
                      to 21.1.2.1 via ge-0/0/1.0, Swap 200104, Push 200105(top)
36(S=0)            *[L-ISIS/14] 02:43:14, metric 0
                   >  to 21.2.4.2 via ge-0/0/3.0, Pop
                      to 21.0.2.1 via ge-0/0/0.0, Swap 200104, Push 200105(top)
                      to 21.1.2.1 via ge-0/0/1.0, Swap 200104, Push 200105(top)
```

Verifying ISIS-SR Forwarding Paths with MPLS Ping and Traceroute

Just as we did for OSPF-SR paths in Chapter 2, we can use MPLS ping and traceroute to verify forwarding using ISIS-SR paths. Even though the L-ISIS routes in inet.3 are not the active route for forwarding due to the presence of more preferred L-OSPF routes, MPLS ping and traceroute can still be used to test forwarding using L-ISIS routes. Specifying `segment-routing isis` with MPLS ping and traceroute commands will cause the command to determine outgoing labels using the L-ISIS routes in inet.3.

You can use MPLS ping to verify the forwarding to reach R7 using the ISIS-SR routes with the following:

```
regress@R0> ping mpls segment-routing isis 7.7.7.7 detail count 1
Request for seq 1, to interface 334, label 200107, packet size 80
Reply for seq 1, return code: Egress-ok, time: 8.723 ms
    Local transmit time: 2021-02-03 16:52:45 PST 406.110 ms
    Remote receive time: 2021-02-03 16:52:45 PST 414.833 ms

--- lsping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
```

Similarly, the next MPLS traceroute command traces out the two ECMP forwarding paths from R0 to R7, using ISIS-SR route entries:

```
regress@R0> traceroute mpls segment-routing isis 7.7.7.7
  Probe options: ttl 64, retries 3, wait 10, paths 16, exp 7, fanout 16

  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           1   200107  ISIS     21.0.2.2       (null)         Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           2   200107  ISIS     21.2.4.2       21.0.2.2       Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           3     3  ISIS     21.4.7.2       21.2.4.2       Egress
  FEC-Stack-Sent: ISIS

  Path 1 via ge-0/0/1.0 destination 127.0.0.64

  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           1   200107  ISIS     21.0.3.2       (null)         Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           2   200107  ISIS     21.3.5.2       21.0.3.2       Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           3     3  ISIS     21.5.7.2       21.3.5.2       Egress
  FEC-Stack-Sent: ISIS

  Path 2 via ge-0/0/2.0 destination 127.0.1.64
```

As was shown for OSPF-SR in Chapter 2, you can run MPLS ping and traceroute commands with an explicit label stack to test the TI-LFA backup path for an ISIS-SR route. As before, originating an MPLS ping or traceroute with an explicit label stack requires `set protocols source-packet-routing`.

The label stack for the TI-LFA backup path to reach R2 can been seen from:

```
regress@R0> show route 2.2.2.2 table inet.3 protocol isis

inet.3: 9 destinations, 23 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2.2.2.2/32       [L-ISIS/14] 02:46:38, metric 1
                 >  to 21.0.2.2 via ge-0/0/1.0
                   to 21.0.3.2 via ge-0/0/2.0, Push 200102, Push 200101(top)
```

This information can be used to construct the following MPLS traceroute command:

```
regress@R0> traceroute mpls segment-routing spring-te label-stack labels [200102 200101] nexthop-
interface ge-0/0/2.0 nexthop-address 21.0.3.2
  Probe options: retries 3, exp 7

  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           1   200101  Static   21.0.3.2       (null)         Success
```

```
FEC-Stack-Sent: SPRING-TE
ttl      Label  Protocol  Address        Previous Hop   Probe Status
         1     3  Static    21.1.3.1       21.0.3.2       Success
FEC-Stack-Sent: SPRING-TE
ttl      Label  Protocol  Address        Previous Hop   Probe Status
         2     3  Static    21.1.2.2       21.1.3.1       Egress
FEC-Stack-Sent: SPRING-TE

Path 1 via ge-0/0/2.0 destination 127.0.0.64
```

Note that MPLS traceroute using an explicit label-stack does not currently support verifying all ECMP paths.

Chapter 5

Shifting Traffic From OSPF-SR to ISIS-SR

Now that you have configured ISIS-SR and verified that the ISIS-SR routes can actually forward traffic, you are ready to shift the traffic from OSPF-SR to ISIS-SR. You can apply the same mechanisms discussed in Chapter 3 [chapter_shifting_ldp_ospf] in this case, too. You can shift traffic from OSPF-SR to ISIS-SR in a very controlled manner using route preference or using multiple loopback addresses.

Shifting Service Traffic from OSPF-SR to ISIS-SR Using Route Preference

In Junos, OSPF and L-OSPF routes have a preference of 10 by default. ISIS level 2 routes have a preference of 18, while L-ISIS routes have a preference of 14:

```
regress@R0> show route 7.7.7.7

inet.0: 48 destinations, 79 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32         *[OSPF/10/10] 00:16:05, metric 3
                   >  to 21.0.2.2 via ge-0/0/1.0
                      to 21.0.3.2 via ge-0/0/2.0
                   [IS-IS/18] 00:15:13, metric 3
                   >  to 21.0.2.2 via ge-0/0/1.0
                      to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 14 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32         *[L-OSPF/10/5] 00:16:05, metric 3
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
```

```
                    to 21.0.3.2 via ge-0/0/2.0, Push 400107
                  [L-ISIS/14] 00:15:13, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                    to 21.0.3.2 via ge-0/0/2.0, Push 200107
```

At this point, a BGP route will resolve on the L-OSPF route (since it has the lowest preference value.) In order to resolve on the L-ISIS routes, you can set the route preference for L-ISIS level 2 routes to 7 using the following configuration on R0:

```
set protocols isis level 2 labeled-preference 7
```

This makes the L-ISIS route have the lowest preference value among the four routes for 7.7.7.7 in inet.0 and inet.3, as can be seen in the following output:

```
regress@R0> show route 7.7.7.7

inet.0: 48 destinations, 79 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32        *[OSPF/10/10] 00:33:59, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0
                    to 21.0.3.2 via ge-0/0/2.0
                  [IS-IS/18] 00:33:07, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0
                    to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 14 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32        *[L-ISIS/7] 00:00:03, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                    to 21.0.3.2 via ge-0/0/2.0, Push 200107
                  [L-OSPF/10/5] 00:33:59, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 400107
                    to 21.0.3.2 via ge-0/0/2.0, Push 400107
```

You can see that the BGP service route advertised by R7 is using the labels provided by L-ISIS:

```
regress@R0> show route 131.1.1.0/24

inet.0: 45 destinations, 73 routes (45 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

131.1.1.0/24      *[BGP/170] 00:33:50, localpref 100, from 7.7.7.7
                    AS path: I, validation-state: unverified
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                    to 21.0.3.2 via ge-0/0/2.0, Push 200107
```

WARNING Changing route preference is a powerful tool in migrations involving segment routing, but it should be used very carefully. Section *LDP OSPF Route Preference* has a detailed discussion of this topic.

Before moving on to the next section, let's make sure to undo changes we made to the default route preference:

```
delete protocols isis level 2 labeled-preference
```

Shifting Service Traffic from OSPF-SR to ISIS-SR Using Multiple Loopback Addresses

You can also shift traffic from OSPF-SR to ISIS-SR using multiple loopbacks as discussed in *LDP OSPF Loopbacks*. The only additional thing you need to understand is how to configure ISIS to advertise a particular loopback address as a node segment. The configuration for ISIS on R7 is shown here:

```
set routing-options static route 71.71.71.71/32 discard

set policy-options policy-statement lb71_sid term one from protocol static
set policy-options policy-statement lb71_sid term one from route-filter 71.71.71.71/32 exact
set policy-options policy-statement lb71_sid term one then prefix-segment index 707
set policy-options policy-statement lb71_sid term one then prefix-segment node-segment
set policy-options policy-statement lb71_sid term one then accept

set protocols isis export lb71_sid
```

The only difference from the OSPF configuration is that the ISIS configuration reuses the existing export statement. After applying the configuration below on R7 to change the BGP next hop:

```
set routing-options static route 131.2.1.0/24 discard

set policy-options policy-statement bgp_export_policy term one from protocol static
set policy-options policy-statement bgp_export_policy term one from route-
filter 131.2.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term one then next-hop 71.71.71.71
set policy-options policy-statement bgp_export_policy term one then accept
```

We get the following output on R0:

```
regress@R0> show route 131.2.1.0/24

inet.0: 49 destinations, 79 routes (49 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

131.2.1.0/24       *[BGP/170] 00:25:20, localpref 100, from 7.7.7.7
                      AS path: I, validation-state: unverified
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200707
                      to 21.0.3.2 via ge-0/0/2.0, Push 200707
```

You can see that the route to reach 131.2.1.0/24 uses label 200707, corresponding to the ISIS SRGB of 200000 and the node-SID index of 707 advertised by ISIS for 71.71.71.71/32.

Chapter 6

Supporting Legacy LDP-only Nodes In an SR Network

In previous chapters we showed how to migrate traffic from OSPF-LDP to OSPF-SR and from OSPF-SR to ISIS-SR. It is straightforward to apply the same techniques of route preference and multiple loopbacks to support a migration from ISIS-LDP to ISIS-SR. In this chapter we'll focus on a partial migration from ISIS-LDP to ISIS-SR, with the limitation that some nodes in the network will not support ISIS-SR for the foreseeable future.

SR/LDP Mapping Servers

The most generally applicable way to support a mixture of LDP-only and ISIS-SR nodes is using SR-LDP mapping servers. In the mapping server approach, one or more routers originate mapping server advertisements that tell ISIS-SR capable routers what prefix-SID value to use for a given prefix. The heavy lifting with respect to forwarding is done by the SR/LDP border nodes, which create label stitching entries in both directions between SR and LDP domains. We do not discuss mapping server in this book. For more detail on the mapping server approach, please consult *Day One: Inside Segment Routing*, written by Anurag Khare and Colby Barth: https://www.juniper.net/documentation/en_US/day-one-books/DO_InsideSR.zip

Coexistence of LDP-only and LDP/SR Nodes Using Route Preference

For some networks, it may make sense to consider a different approach to the co-existence of SR and LDP in the network. The main idea is to continue to use LDP forwarding from end-to-end when at least one of the two service endpoints does not support SR. The main restriction in terms of topology is that there should not be any transit traffic on an LDP-only node. Figure 6.1 [fig_ldp_sr_coexistence] shows a version of our example network with the metrics on the links involving R6 and R7 increased to 1000, so that there will be no transit traffic on R6 and R7.

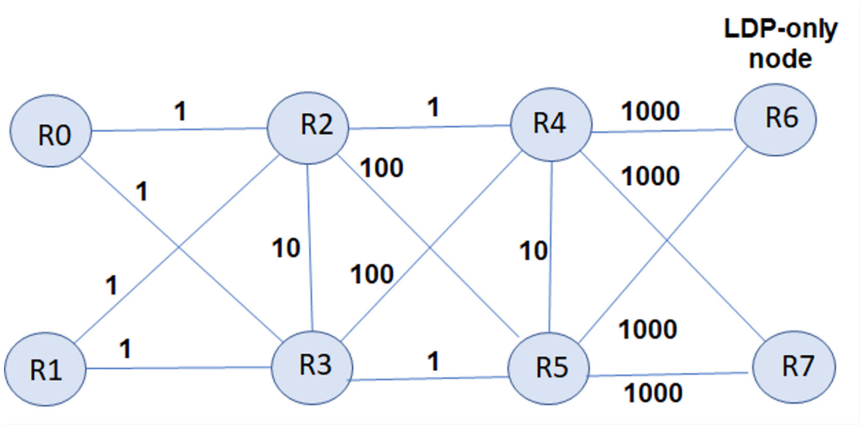


Figure 6.1 Example Topology with Modified Metrics

You can see the modified metrics prevent transit traffic on R6 and R7. R6 only runs LDP, while all other nodes run both LDP and SR. To see how this solution works in practice, let's remove any existing OSPF configuration from all nodes and activate the basic LDP configuration on all nodes. Make sure you have applied the basic ISIS-SR configuration from Chapter 4[chapter_ospf_sr_isis_sr] on all nodes.

To illustrate the kind of topology where LDP can safely coexist with ISIS-SR without a mapping server, configure the metrics to be 1000 on both sides of R4-R6, R4-R7, R5-R6, and R5-R7, so that R6 and R7 are not expected to carry any transit traffic:

```
R4# set protocols isis interface ge-0/0/5.0 level 2 metric 1000
R4# set protocols isis interface ge-0/0/6.0 level 2 metric 1000
R5# set protocols isis interface ge-0/0/5.0 level 2 metric 1000
R5# set protocols isis interface ge-0/0/6.0 level 2 metric 1000
R6# set protocols isis interface ge-0/0/5.0 level 2 metric 1000
R6# set protocols isis interface ge-0/0/6.0 level 2 metric 1000
R7# set protocols isis interface ge-0/0/5.0 level 2 metric 1000
R7# set protocols isis interface ge-0/0/6.0 level 2 metric 1000
```

We want to simulate the situation where R6 does not support ISIS-SR. R6 only runs ISIS with LDP. Do this with the following configuration on R6:

```
R6# deactivate protocols isis source-packet-routing
```

With the default route preference values for LDP, ISIS and labeled ISIS on R0, you can see the following routes in inet.0 and inet.3 for 6.6.6.6 and 7.7.7.7:

```
regress@R0> show route 6.6.6.6

inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

6.6.6.6/32         *[IS-IS/18] 00:33:23, metric 12
                   >  to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 13 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

6.6.6.6/32         *[LDP/9] 00:27:03, metric 1
                   >  to 21.0.3.2 via ge-0/0/2.0, Push 30

[edit]
regress@R0> show route 7.7.7.7

inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32         *[IS-IS/18] 00:33:32, metric 1002
                   >  to 21.0.2.2 via ge-0/0/1.0
                      to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 13 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32         *[LDP/9] 00:27:12, metric 1
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 36
                      to 21.0.3.2 via ge-0/0/2.0, Push 39
                    [L-ISIS/14] 00:00:22, metric 1002
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                      to 21.0.3.2 via ge-0/0/2.0, Push 200107
```

At this point, BGP routes with protocol next-hop 6.6.6.6 and 7.7.7.7 should both be resolving on LDP routes in inet.3. The following output on R0 confirms this:

```
regress@R0> show route protocol bgp table inet.0

inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

121.1.1.0/24       *[BGP/170] 00:26:29, localpref 100, from 6.6.6.6
                      AS path: I, validation-state: unverified
                   >  to 21.0.3.2 via ge-0/0/2.0, Push 30
```

```
121.2.1.0/24      *[BGP/170] 00:26:29, localpref 100, from 6.6.6.6
                    AS path: I, validation-state: unverified
                  >  to 21.0.3.2 via ge-0/0/2.0, Push 30
131.1.1.0/24      *[BGP/170] 00:06:02, localpref 100, from 7.7.7.7
                    AS path: I, validation-state: unverified
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 36
                    to 21.0.3.2 via ge-0/0/2.0, Push 39
131.2.1.0/24      *[BGP/170] 00:06:02, localpref 100, from 7.7.7.7
                    AS path: I, validation-state: unverified
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 36
                    to 21.0.3.2 via ge-0/0/2.0, Push 39
```

You can start using ISIS-SR paths, where they exist, by changing the labeled ISIS route preference on R0 for L2 routes from 14 to 7:

```
R0# set protocols isis level 2 labeled-preference 7
```

This causes BGP to prefer ISIS-SR routes over LDP routes when both types of routes go to a destination, but still use LDP routes when ISIS-SR routes are not available. The effect of this change can be seen in the following output on R0:

```
regress@R0> show route 6.6.6.6

inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

6.6.6.6/32        *[IS-IS/18] 01:47:48, metric 12
                  >  to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 13 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

6.6.6.6/32        *[LDP/9] 01:41:28, metric 1
                  >  to 21.0.3.2 via ge-0/0/2.0, Push 30

regress@R0> show route 7.7.7.7

inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32        *[IS-IS/18] 01:47:54, metric 1002
                  >  to 21.0.2.2 via ge-0/0/1.0
                    to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 13 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

7.7.7.7/32        *[L-ISIS/7] 00:03:07, metric 1002
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                    to 21.0.3.2 via ge-0/0/2.0, Push 200107
                   [LDP/9] 01:41:34, metric 1
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 36
                    to 21.0.3.2 via ge-0/0/2.0, Push 39
```

```
regress@R0> show route protocol bgp table inet.0

inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

121.1.1.0/24       *[BGP/170] 01:37:09, localpref 100, from 6.6.6.6
                      AS path: I, validation-state: unverified
                   >  to 21.0.3.2 via ge-0/0/2.0, Push 30
121.2.1.0/24       *[BGP/170] 01:37:09, localpref 100, from 6.6.6.6
                      AS path: I, validation-state: unverified
                   >  to 21.0.3.2 via ge-0/0/2.0, Push 30
131.1.1.0/24       *[BGP/170] 00:05:05, localpref 100, from 7.7.7.7
                      AS path: I, validation-state: unverified
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                      to 21.0.3.2 via ge-0/0/2.0, Push 200107
131.2.1.0/24       *[BGP/170] 00:05:05, localpref 100, from 7.7.7.7
                      AS path: I, validation-state: unverified
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                      to 21.0.3.2 via ge-0/0/2.0, Push 200107
```

The solution illustrated above allows LDP-only nodes to coexist with nodes that run both ISIS-SR and LDP. It is only applicable when the nodes in the LDP-only domain are not expected to carry traffic between ISIS-SR domains. However, due to its simplicity of deployment, it is worth considering in some network migration scenarios.

Before moving on to the Chapter 7, remember to undo the temporary metric changes that were made to the example topology and remember to activate ISIS-SR on R6.

Chapter 7

A Deep Dive Into SRGBs

SRGB is a range of label values reserved for segment routing. SRGB and index values are used to derive the actual label values for a prefix. Segment routing allows for SRGB to be different on every node. This flexibility allows for the possibility of different hardware platforms using different SRGB in the same network.

In Junos 17.2 and onwards, SRGB configuration is supported. The previous versions used local label block configurations that dynamically allocated label blocks for segment routing. This mechanism had challenges such as the dynamic label block allocation varying on every restart, and the label block allocated to SR was not deterministic.

Now that the label manager is supported in 17.2 onwards, the label manager does not make any hard partitioning of label types for LSI, block labels, and dynamic labels. On platforms that do not specify fixed LSI labels, the entire label range of 16-999999 is available for all the applications. Effectively, this means that any block from this 16-999999 range can be used for SRGB.

There are various advantages of configuring SRGB:

- The label block is deterministic and does not change.
- Nodes assigned with anycast-SID can use the same SRGB. This helps with simplified SR label stack construction.
- Deterministic labels help with easier debugging in case of problems in the network.

The SRGB range that is allowed on different Juniper platforms is:

- MX: 16-999999
- PTX: 16-999999
- ACX 5448/ACX 710: 16-991807
- QFX: SRGB not qualified

On QFX switches the SRGB feature is not qualified. Basic SR with dynamic block allocation has been qualified. Technically, the QFX 5k range of devices should be able to support 16-999999 label range. Configuring SRGB on a node requires MX and PTX devices to be running in enhanced services mode. When this mode is enabled for the first time, the router must be rebooted.

Configuring SRGB for the First Time

Post 17.2 release, which has the new label manager support, the network service enhanced-ip mode should be enabled.

On a router the following command can be used to verify whether the new label manager has been running:

```
regress@R3> show mpls label usage
Label space Total   Available     Applications
LSI        69609   69609  (100.00%) BGP/LDP VPLS with no-tunnel-services, BGP L3VPN with vrf-table-
label
Block      199936  199936 (100.00%) BGP/LDP VPLS with tunnel-services, BGP L2VPN
Dynamic    487936  487929 (100.00%) RSVP, LDP, PW, L3VPN, RSVP-P2MP, LDP-P2MP, MVPN, EVPN, BGP
Static     48576   48576  (100.00%) Static LSP, Static PW
```

This output does not show the effective label blocks for LSI, Block, and Dynamic. This implies that the new label manager has not been running.

Enable new the label manager by configuring `enhanced-ip` mode.

set chassis network-services enhanced-ip

Configure SRGB in OSPF:

```
set protocols ospf source-packet-routing srgb index-range 5000 start-label 400000
```

Without a reboot the label manager will remain in the old, partitioned mode and the SRGB allocation will fail:

```
[edit]
regress@R3# run show ospf overview
Instance: master
  Router ID: 3.3.3.3
  Route table index: 0
  LSA refresh time: 50 minutes
  SPRING: Enabled
          SRGB Config Range :
```

```
         SRGB Start-Label : 400000, SRGB Index-Range : 5000
         SRGB Block Allocation: Failure
         SRGB Start Index : 400000, SRGB Size : 5000, Label-Range: [ 400000, 404999 ]
         Node Segments: Enabled
         Ipv4 Index : 103
  Post Convergence Backup: Disabled
  Area: 0.0.0.0
         Stub type: Not Stub
         Authentication Type: None
         Area border routers: 0, AS boundary routers: 0
         Neighbors
         Up (in full state): 5
  Topology: default (ID 0)
         Prefix export count: 0
         Full SPF runs: 9
         SPF delay: 0.200000 sec, SPF holddown: 5 sec, SPF rapid runs: 3
         Backup SPF: Not Needed
```

Re-verification of MPLS label usage implies that the device has not been rebooted:

```
regress@R3# run show mpls label usage
Label space Total   Available     Applications
LSI        69609   69609  (100.00%) BGP/LDP VPLS with no-tunnel-services, BGP L3VPN with vrf-table-
label
Block      199936  199936 (100.00%) BGP/LDP VPLS with tunnel-services, BGP L2VPN
Dynamic    487936  487924 (100.00%) RSVP, LDP, PW, L3VPN, RSVP-P2MP, LDP-P2MP, MVPN, EVPN, BGP
Static     48576   48576  (100.00%) Static LSP, Static PW
```

After configuring `enhanced-ip` mode, the router needs to be rebooted. Go ahead and reboot the device and verify the label usage:

```
regress@R0> show mpls label usage
Label space Total   Available     Applications
LSI        999984  999977 (100.00%) BGP/LDP VPLS with no-tunnel-services, BGP L3VPN with vrf-table-
label
Block      999984  999977 (100.00%) BGP/LDP VPLS with tunnel-services, BGP L2VPN
Dynamic    999984  999977 (100.00%) RSVP, LDP, PW, L3VPN, RSVP-P2MP, LDP-P2MP, MVPN, EVPN, BGP
Static     48576   48576  (100.00%) Static LSP, Static PW
Effective Ranges
Range name  Shared with Start    End
Dynamic                 16       999999
Static                  1000000  1048575
Configured Ranges
Range name  Shared with Start   End
Dynamic                 16      999999
Static                  1000000 1048575
```

Note the differences when the new configurable label manager is in use as compared to the legacy label manager. The new label manager has the effective configurable label range which is available from 16-999999 as indicated under "Effective Ranges." SRGB blocks can be picked from within this block.

Verify that the OSPF SRGB allocation has been successful:

```
regress@R3# run show ospf overview
Instance: master
  Router ID: 3.3.3.3
  Route table index: 0
  LSA refresh time: 50 minutes
  SPRING: Enabled
           SRGB Config Range :
           SRGB Start-Label : 400000, SRGB Index-Range : 5000
           SRGB Block Allocation: Success
           SRGB Start Index : 400000, SRGB Size : 5000, Label-Range: [ 400000, 404999 ]
           Node Segments: Enabled
           Ipv4 Index : 103
  Post Convergence Backup: Disabled
  Area: 0.0.0.0
           Stub type: Not Stub
           Authentication Type: None
           Area border routers: 0, AS boundary routers: 0
           Neighbors
           Up (in full state): 5
  Topology: default (ID 0)
           Prefix export count: 0
           Full SPF runs: 9
           SPF delay: 0.200000 sec, SPF holddown: 5 sec, SPF rapid runs: 3
           Backup SPF: Not Needed
```

When there are dynamic label allocation protocols such as LDP, RSVP, and BGP-LU running in the network, these protocols might have allocated labels from the older dynamic label range.

SRGB allocation will be successful only when the configured SRGB does not conflict with any of the allocated labels. It is recommended to set the node for overload and disable NSR and graceful restart for LDP, RSVP, and BGP before the reboot. This will ensure that when the router reboots it will first reserve the configured SRGB and then allocate labels for the dynamic label protocol outside of SRGB range. If NSR/graceful restart is enabled, the individual protocol procedures may try to allocate the previously allocated labels, which may lead to conflicts.

Operational Considerations

Shared SRGB vs Per-Protocol SRGB

SRGB can be configured in two different ways: shared SRGB and per-protocol SRGB. Shared SRGB can be configured under `protocols mpl`:

```
set protocols mpls  label-range srgb-label-range <range-start> <range-end>
```

Every protocol has a separate SRGB configuration under the `protocols <x>` stanza:

```
set protocols ospf source-packet-routing srgb index-range 5000 start-label 400000
```

When the per-protocol SRGB is configured, all protocols like OSPF/ISIS/BGP advertise the SRGB from the configured value under each protocol stanza.

If a shared SRGB is configured under the protocol MPLS stanza, every protocol will use the same SRGB in its protocol advertisements. When both shared and per-protocol SRGB are configured, the per-protocol SRGB will override the shared SRGB. If a shared SRGB is used, the operator has to ensure the indices for prefixes do not overlap across protocols.

If there is a conflict within the protocol, protocol mechanisms will detect the conflict but if the conflict is across protocols such a conflict is not easily detected and reported.

Per-protocol SRGB is very useful when the networks are getting merged or migrated. If two networks that were under different administration running two different IGPs come under single administration, per-protocol SRGB configuration can be used on common nodes to allow for merging the network while still maintaining the individual IGP configurations in each domain.

Size of SRGB

Choosing the size of the SRGB is an important planning exercise. The size should be chosen in consideration of future network growth. Changing SRGB configurations requires the router to be moved to a maintenance state. Repeated changes to the SRGB size should be avoided in order to reduce maintenance windows. Keeping SRGB uniform across the network has advantages so operators should account for SRGB range supported on all vendors in a multi-vendor network. Juniper MX and PTX devices have opened up label space from 16-999999 and are most flexible in terms of SRGB range availability.

SRGB Configuration Without Reboot

In certain networks, there may be a need to configure the SRGB without requiring a reboot.

It is possible to configure SRGB without reboot if the following conditions are met:

- The router is running post 17.2 version with the new label manager in force.
- Verify `show mpls label usage` to confirm the router has already been rebooted since the new label manager was turned on and the new label manager is effective.
- Ensure no other label allocation protocol is running.
- If these conditions are met, SRGB can be safely configured without requiring reboot.

If there are other dynamic label allocation protocols like LDP, RSVP and BGP-LU , OSPF-SR, or ISIS-SR running on the router, then the operator is strongly recommended to reboot the device to avoid conflicts. If the operator can ensure a certain label range is available and not conflicting with any of the already allocated labels, then SRGB can be configured without reboot.

In case SRGB allocation fails due to conflicts, operators can clear the conflicting labels by deactivating the protocol or adjacency for which the conflicting label is allocated. After freeing the conflicting labels, the below command can be used to re-trigger SRGB allocation:

```
request mpls label-range reserve
```

Chapter 8

Segment Routing Traffic Counters

SR traffic counters provide the ability to build a traffic matrix for SR traffic. SR traffic counters can be reaped from the routers with streaming telemetry. The details on how to configure sensors in ISIS and use jtimon collector to collect the traffic counters is explained in detail within *Day One: Inside Segment Routing*, https://www.juniper.net/documentation/en_US/day-one-books/DO_InsideSR.zip.

When the network is being migrated to SR, the SR counters will provide details of the SR-specific traffic, and the LDP or RSVP traffic will be counted separately by respective protocol counters. In this chapter we will learn some details about the type of counters supported in ISIS SR. We will also see how these counters can be used to build the traffic matrix. We will discuss some challenges typically faced with SR. Figure 8.1 illustrates the lab topology with ISIS-SR enabled as well as the SRGB and the indices.

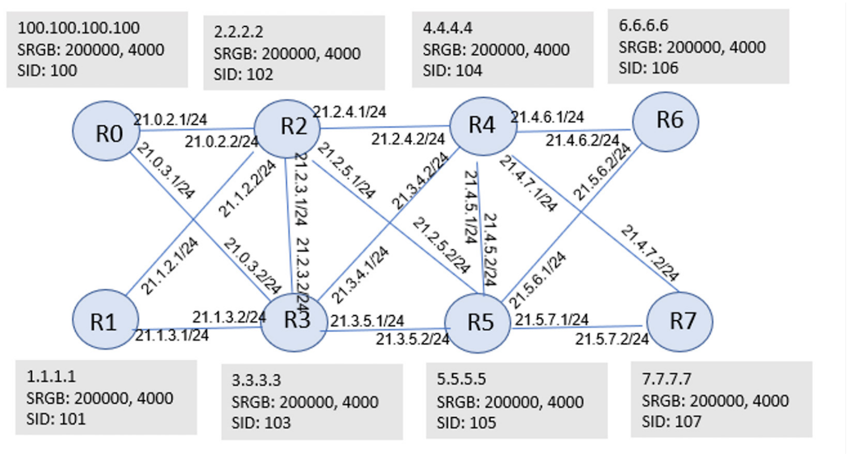


Figure 8.1 Sample SR Network

Junos ISIS SR supports below types of counters:

- Per-SID ingress
- Per-SID egress
- Per-interface-per-member-link egress
- Per-interface-per-member-link ingress

Per-SID ingress counters provide the transit MPLS traffic towards a particular destination on a particular router. The counters are identified by the label towards the destination. For example, on R0, the transit traffic towards R7 counted in the per-SID ingress counter named “200107”. The label is derived based on the SRGB configured on that router. In the lab topology, the same SRGB is configured on all nodes. Suppose R3 is configured with SRGB of start label 600000 and size 5000, the transit traffic on this router R3 will be identified with a sensor with name *600107*. When there are ECMP next hops to 7.7.7.7, aggregate traffic towards 7.7.7.7 will be counted in this sensor. Fine-grained counters such as per-SID-per-Interface sensors are not supported at the time of writing this book.

Per-SID egress counters provide the traffic counters for the IP traffic that gets encapsulated in MPLS headers on the routers. This is referred to as IP->MPLS traffic. These counters are identified with the IP address of the remote PE to which the MPLS encapsulation is done. For example, let’s take a case of traffic going from R0 to R7. R0 is an ingress router that encapsulates incoming IP traffic into MPLS. The traffic for R7 is counted using the per-SID egress counter named “L-ISIS-7.7.7.7”. Similarly, traffic headed to 6.6.6.6 is counted using the sensor named “L-ISIS-6.6.6.6”. Similar to per-SID ingress counters, ECMP traffic is counted in aggregate and per-SID-per-interface granularity counters are not supported.

In order to build the demand matrix from R0 to R7, the per-SID egress counter “L-ISIS-7.7.7.7” and per-SID ingress counter for “200107” should be added together to get total traffic.

Per-interface-per-memberlink egress counter provides total outgoing SR traffic on an interface. Similarly, the per-interface-per-memberlink ingress counter provides total incoming SR traffic on an interface. The counters are streamed on a per member-link granularity in case of an AE bundle link. The per-interface-per-memberlink egress counter is useful during migrations. It is useful in determining the total SR traffic on an interface and feed that consumed traffic into “available traffic” from RSVP. The per-interface-per-memberlink egress counter is used to collect total SR traffic when the SR-auto bandwidth feature is enabled.

Per-interface-per-memberlink ingress counters provide total incoming SR traffic on an interface. The traffic is counted on a per-memberlink granularity for an AE bundle link. This counter can be used to compare against the per-interface-per-memberlink egress counter and can be used to detect SR traffic drops.

Some operators measure the per-VRF traffic on internal P devices using netflow. Netflow makes use of the firewall filters to look up the bottom-most MPLS label in the packet or to look up the IP headers of the original packet and count the traffic for the VRF. In the case of LDP/RSVP, the position of the VPN label from the top of the MPLS header is fixed in most cases. In the case of SR, the position of VPN labels may not be fixed if FRR protection is in force. TI-LFA can impose arbitrary numbers of labels so the total number of MPLS labels on the packet is not fixed. Netflow firewall filters may not be able to read past the arbitrary number of labels to reach VPN labels. In certain cases, if the number of labels Netflow has to read is beyond a certain number, that may not be supported.

In the case of SR networks, it is recommended to measure the VPN traffic on CE-PE interfaces.

Basic Show Command to Show Counters On PFE

Let’s enable traffic counting and verify the counters on the PFE.

In the lab topology shown in Figure 8.1, you need to find the traffic flowing between R0 to R1, R6, and R7. This is basically finding traffic between each pair of PEs. The SID allocated for R6 is 106. The label on R0 for R6 is 200106. On R0, the counters allocated for the Prefix 6.6.6.6 identified by sensor name “L-ISIS-6.6.6.6” will give traffic towards the R6 router.

Verify that the route 121.1.1.1 on R0 is resolving on an ISIS-SR route to R6:

```
regress@R0# run show route 121.1.1.1

inet.0: 78 destinations, 105 routes (78 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

121.1.1.0/24      *[BGP/170] 03:11:45, localpref 100, from 6.6.6.6
                    AS path: I, validation-state: unverified
                    to 21.0.2.2 via ge-0/0/1.0, Push 200106
                  >  to 21.0.3.2 via ge-0/0/2.0, Push 200106
```

Ping packets originated from R0 will not be accounted for statistics. Data packets that hit the next hops in the PFE will be accounted.

For this exercise let’s temporarily modify the routes to simulate data traffic on R0. On R2 create a static route for 121.1.1.1 to point towards R0 and force the data packets through R0 to verify the counters.

From R2, send 10 packets for prefix 121.1.1.1 via R0.

When traffic is received on R0, it is still an IP packet and will hit the BGP route that resolves an ISIS-SR route:

```
regress@R2> ping rapid 121.1.1.1 count 10
PING 121.1.1.1 (121.1.1.1): 56 data bytes
!!!!!!!!!
--- 121.1.1.1 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 4.423/14.691/187.289/20.480 ms
```

On R0, let us look at the IP->MPLS counters for the sensor L-ISIS-6.6.6.6. We do not have CLI support to display the statistics, but we can directly check the counters by logging into the PFE. This mechanism is not recommended for a production environment but can be used for debugging purposes in a lab:

```
regress@R0>request pfe execute command "show agent sensor" target fpc0 | grep "ip-mpls" |no-more

 2733067753  ip-mpls                5000(203)


regress@R0# request pfe execute command "show agent sensor id 2733067753" target fpc0 | no-more


SENT: Ukern command: show agent sensor id 2733067753

ID: 2733067753 Type: 2 Name: (ip-mpls)
Sensor Data: 229(201) bytes
Jvision Header: system: shraddha-r0-r:100.1.1.1, slot: 0, time: Jun  9 09:25:01.386, sequence_
number: 1, sensor_name: ip-mpls:/junos/services/segment-routing/sid/egress/usage/:/junos/services/
segment-routing/sid/egress/usage/:PFE, version: 1.0
---------------------------------------------------------------------------------------
           Packets        Bytes Packet RateByte Rate Inst  Counter-Name SID Name
--------------- --------------- ----------- ------------ ---- ------------- --------
           10          840       0       810 oc-17  L-ISIS-6.6.6.6
---------------------------------------------------------------------------------
Total 1 records.
---------------------------------------------------------------------------------
           Export: Interval = 5000 msecs, Payload-size = 5000 bytes, Server = Src: 100.1.1.1:1000, Dst:
100.1.1.2:2000(VRF 0) , Qos = TOS:0x0, Hint:0x40000010, FC:0 PLP:0
           Filter: none
           Accounting:
           63287 total successful reaps, 0 failed/aborted reap(s).
               1 reaps in latest reporting interval.
               1 packets in latest reporting interval.
               1 instances in latest reporting interval.
               2 total packets sent.
           63287 wraps, 1 reap(s) on average to wrap.
           Last 10 wraps (UTC):
             2020 Jun 09 09:25:01:386
             2020 Jun 09 09:24:56:262
             2020 Jun 09 09:24:51:126
             2020 Jun 09 09:24:45:980
             2020 Jun 09 09:24:40:847
             2020 Jun 09 09:24:35:706
             2020 Jun 09 09:24:30:562
             2020 Jun 09 09:24:25:432
             2020 Jun 09 09:24:20:301
             2020 Jun 09 09:24:15:176
---------------------------
```

You can see that the PFE shows ten packets for the sensor “L-ISIS-6.6.6.6” on which the traffic is flowing.

During migration, if multiple loopbacks are used on the remote router, per-SID egress counters for both loopback addresses should be added together to get the aggregate traffic from R1 to R6

There may be some transit traffic going towards R6 on R0.

This traffic is accounted as part of the per-SID ingress counter.

For this exercise, we temporarily made some topology modifications to simulate transit traffic towards R6 to go via R0. On R2 we brought down interfaces towards R3, R4, and R5 and forced the traffic with destination R6 to go towards R0.

Now we enable `bgp-igp-both-ribs` on R2 to place L-ISIS routes in inet.0. This is required to enable ping to use L-ISIS routes in inet.0:

```
regress@R0> set protocols mpls traffic-engineering bgp-igp-both-ribs

[edit]
regress@R2# run show route 6.6.6.6

inet.0: 66 destinations, 73 routes (66 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

6.6.6.6/32       *[L-ISIS/14] 00:05:24, metric 4
                 >  to 21.0.2.1 via ge-0/0/0.0, Push 200106
                    to 21.1.2.1 via ge-0/0/1.0, Push 200106
                 [IS-IS/18] 00:05:24, metric 4
                 >  to 21.0.2.1 via ge-0/0/0.0
                    to 21.1.2.1 via ge-0/0/1.0
```

Ping packets are sent to 6.6.6.6 and on R0 you can see transit traffic for label 200106:

```
regress@R2# run ping 6.6.6.6
PING 6.6.6.6 (6.6.6.6): 56 data bytes
64 bytes from 6.6.6.6: icmp_seq=0 ttl=61 time=6.615 ms
64 bytes from 6.6.6.6: icmp_seq=1 ttl=61 time=7.244 ms
64 bytes from 6.6.6.6: icmp_seq=2 ttl=61 time=15.247 ms
64 bytes from 6.6.6.6: icmp_seq=3 ttl=61 time=8.364 ms
64 bytes from 6.6.6.6: icmp_seq=4 ttl=61 time=9.919 ms
^C
--- 6.6.6.6 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 6.615/9.478/15.247/3.095 ms



inet.3: 7 destinations, 8 routes (7 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
6.6.6.6/32       *[L-ISIS/14] 00:05:24, metric 4
                 >  to 21.0.2.1 via ge-0/0/0.0, Push 200106
                    to 21.1.2.1 via ge-0/0/1.0, Push 200106


regress@R0> request pfe execute command "show agent sensor" target fpc0 | grep "per-sid" |no-more
3442950488  per-sid                 5000(0)


regress@R0> request pfe execute command "show agent sensor id 3442950488" target fpc0 | no-more
SENT: Ukern command: show agent sensor id 3442950488

ID: 3442950488 Type: 2 Name: (per-sid)
Sensor Data: 295(267) bytes
Jvision Header: system: shraddha-r0-r:100.1.1.1, slot: 0, time: Jun  9 09:44:57.826, sequence_
number: 74, sensor_name: per-sid:/junos/services/segment-routing/sid/usage/:/junos/services/
segment-routing/sid/usage/:PFE, version: 1.0
------------------------------------------------------------------------------------
          Packets        Bytes Packet RateByte Rate Inst  Counter-Name SID Name
--------------- --------------- ----------- ------------ ---- ------------- --------
          5         320       0       310 oc-7    200103
          4         256       0         0 0 oc-8  200104
          5         320       0         0 0 oc-9  200105
          5         2288      1       128 0 oc-10 200106
------------------------------------------------------------------------------
Total 4 records.
------------------------------------------------------------------------------
          Export: Interval = 5000 msecs, Payload-size = 5000 bytes, Server = Src: 100.1.1.1:1000, Dst:
100.1.1.2:2000(VRF 0) , Qos = TOS:0x0, Hint:0x40000010, FC:0 PLP:0
          Filter: none
          Accounting:
          63904 total successful reaps, 0 failed/aborted reap(s).
              1 reaps in latest reporting interval.
              1 packets in latest reporting interval.
              4 instances in latest reporting interval.
            75 total packets sent.
          63903 wraps, 1 reap(s) on average to wrap.
          Last 10 wraps (UTC):
            2020 Jun 09 09:44:57:827
            2020 Jun 09 09:44:52:690
            2020 Jun 09 09:44:47:560
            2020 Jun 09 09:44:42:433
            2020 Jun 09 09:44:37:290
            2020 Jun 09 09:44:32:154
            2020 Jun 09 09:44:26:973
            2020 Jun 09 09:44:21:848
            2020 Jun 09 09:44:16:705
            2020 Jun 09 09:44:11:583
---------------------------
```

You can see that transit traffic is counted for sensor named 200106. The total traffic for R6 should be derived from adding L-ISIS-6.6.6.6 and the counter for 200106.

Remember to revert the temporary changes done in this chapter before proceeding with Chapter 9.

Chapter 9

Separating Routing Planes with Flex-Algo

In Segment Routing, a prefix segment follows the least-cost path from its source to its destination. The IGP calculates and identifies the least-cost path. A network operator can configure multiple prefix segments ending on the same destination. Flexible algorithm (flex-algo) is a mechanism that allows network operators to influence how the IGP calculates the least cost path for each prefix segment. Therefore, each prefix segment can traverse a unique path to the destination. This chapter explores how flex-algo supports Traffic Engineering (TE) in our network.

Those who are less familiar with flex-algo might find the following resources useful:

- *Day One: Inside Segment Routing*, written By Anurag Khare and Colby Barth: https://www.juniper.net/documentation/en_US/day-one-books/DO_InsideSR.zip
- *IGP Flexible Algorithms (Flex-Algo) blog* written by Ron Bonica: https://blogs.juniper.net/en-us/industry-solutions-and-trends/igp-flexible-algorithms-flex-algo

Why Flexible Algorithms?

Dual plane network architecture is one of the common network architectures deployed by network operators. Certain use cases require strict traffic separation along the two planes. Flex-algo is an easy and efficient mechanism to separate routing planes.

In SR, TE paths can be built by stacking node-SIDs and adj-SIDs. The path consisting of only adj-SIDs can be arbitrarily long and hit the label stack depth limitation on the ingress PE router hardware.

Node-SIDs are most commonly used to compress the label stacks. While compressing the label-stacks with node-SIDs, the compute engine ensures that the compressed path also satisfies all the constraints of the TE path. Until the network and the explicit path computation engine converge completely when there is a failure, the node-SIDs may traverse a path that does not satisfy TE constraints.

Many use cases such as data sovereignty require the TE constraints to be strictly met at all times. Certain customer traffic requirements avoid passing through nodes and links located in certain geographical regions and these constraints need to be strictly honored at all times, even when there are failures. Flex-algo solves this use case very efficiently. Flex-algo topology is built with all nodes that advertise a particular flex-algo and satisfy constraints described in flexible algorithm definition (FAD). The paths will be built only within this topology. Nodes and links outside of this topology will not be visible for the flex-algo SPF computation. This mechanism provides a very effective way of isolating the nodes and links. The backup paths computed for the flex-algo routes will also be within the flex-algo topology and hence honor the constraints.

MORE? The details of flex-algo configurations are covered in *Day One: Inside Segment Routing*, https://www.juniper.net/documentation/en_US/day-one-books/DO_InsideSR.zip

This chapter focuses on how to migrate services onto flex-algo. In the later part of this chapter we will also cover some of the operational aspects of flex-algo.

Flex-Algo Configuration

Figure 9.1 depicts the lab topology used for this chapter. The links are configured with a TE metric. Flex-algo 133 is enabled on all nodes. The flex-algo SIDs for R0, R1, and etc., is set to 300, 301, and etc. as shown in the topology diagram of Figure 9.1.

R0 is chosen as the FAD server in this topology and R0 has configurations for the FAD. In a deployed network, it is recommended to configure two or more FAD servers to ensure redundancy. Restricting flex-algo definition to only a few nodes ensures that the other routers only need to configure flex-algo participation. This minimizes the possibility of configuration errors that could arise if the FAD needs to be configured on all nodes in a large network.

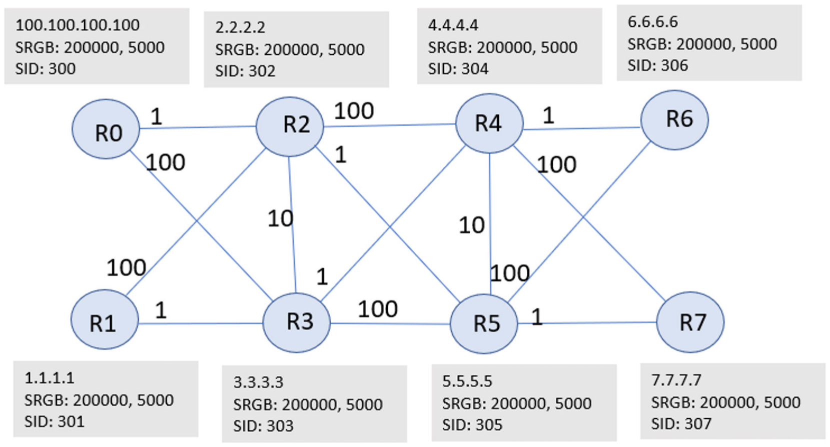


Figure 9.1 Topology Representing TE-metric Configurations

Flex-algo SIDs need to be assigned on all routers as shown here:

```
regress@R0> show configuration policy-options policy-statement flex_sid_policy|display set
set policy-options policy-statement flex_sid_policy term one from route-
filter 100.100.100.100/32 exact
set policy-options policy-statement flex_sid_policy term one then prefix-
segment algorithm 133 index 300
set policy-options policy-statement flex_sid_policy term one then prefix-segment algorithm 133 node-
segment
```

Note that the flex-algo SIDs must have the `node-segment` bit enabled on them as shown in the last line of the configuration sample. This will ensure the TI-LFA compressed paths for the flex-algo routes can use these flex-algo SIDs for compression. The rationale for this is described in detail in section TI_LFA of this chapter.

Flex-algo is enabled on ISIS with the configuration:

```
set protocols isis source-packet-routing flex-algorithm 133
```

With this configuration, ISIS advertises the algorithm sub-TLV for algorithm 133 and the flex-algo SIDs corresponding to 133. If the Node is a FAD server, it also advertises the FAD sub-TLV. All nodes participating in the flex-algo 133 compute the SPF paths based on te-metric (based on the FAD definition) and create MPLS routes.

The transit routes get downloaded into mpls.0 table. The routes in mpls.0 table will get downloaded to the FIB as well. The ingress routes will get created in inetcolor.0 table.

Inetcolor.0 table is a tunnel table similar to inet.3. The routes in inetcolor.0 do not get downloaded to FIB until some service prefix resolves on these routes.

Once flex-algo is configured on all the routers the inetcolor.0 table is built as shown here:

```
regress@R0> show route table inetcolor.0

inetcolor.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1-133<c>/64
                   *[L-ISIS/14] 00:00:54, metric 12
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200301
                     to 21.0.3.2 via ge-0/0/2.0, Push 200301
2.2.2.2-133<c>/64
                   *[L-ISIS/14] 00:00:45, metric 1
                   >  to 21.0.2.2 via ge-0/0/1.0
                     to 21.0.3.2 via ge-0/0/2.0, Push 200302
3.3.3.3-133<c>/64
                   *[L-ISIS/14] 00:20:03, metric 11
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200303
                     to 21.0.3.2 via ge-0/0/2.0
4.4.4.4-133<c>/64
                   *[L-ISIS/14] 00:00:31, metric 12
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200304
                     to 21.0.3.2 via ge-0/0/2.0, Push 200304
5.5.5.5-133<c>/64
                   *[L-ISIS/14] 00:00:27, metric 2
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200305
                     to 21.0.3.2 via ge-0/0/2.0, Push 200305
6.6.6.6-133<c>/64
                   *[L-ISIS/14] 00:00:22, metric 13
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200306
                     to 21.0.3.2 via ge-0/0/2.0, Push 200306
7.7.7.7-133<c>/64
                   *[L-ISIS/14] 00:00:18, metric 3
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200307
                     to 21.0.3.2 via ge-0/0/2.0, Push 200307
```

Note that the routes in inetcolor.0 are suffixed with the color 133 and the prefix length is 64 bits. By default, flex-algo routes use the algorithm number as color. The color can be modified with this configuration in the `flex-algorithm` definition:

```
set routing-options flex-algorithm 133 color 100
```

Deployed services may be already associated with extended color communities. When flex-algo is introduced into the network, the ability to modify the flex-algo route color to match with already deployed services is very useful. This way the services prefixes with color community 100 can map onto those using flex-algo paths with matching color 100.

Verifying Flex-Algo Paths

Flex-algo paths can be verified with MPLS ping and traceroute using label stacks of flex-algo SIDs. This mechanism uses NIL FEC and does not perform any validation on the transit routers for traceroute. It's useful in cases where intermediate routers are not upgraded to support FEC validation.

NIL FEC is defined in RFC 8029 where it is used in MPLS OAM for implicit null and explicit null labels. In the case of SR, NIL FEC is used to traverse the path without validation for cases where the FEC is not defined for newer features. A detailed procedure of how this is used can be found in *draft-rathi-mpls-egress-tlv-for-nil-fec*.

Let's verify the flex-algo path to R4. Flex-algo node-SID for R4 is 304. Label 200304 is derived for R4 based on the SRGB of R0. A single label 200304 is added in the label stack for the ping and traceroute commands. For `label-stack` based ping and traceroute, the next-hop address and next-hop interface need to be specified explicitly in the command.

Here's an example of MPLS ping to R4 with 2000304 in the label-stack showing success indicating the forwarding path to R4 is up and running:

```
regress@R0>ping mpls segment-routing spring-te label-stack labels 200304 nexthop-
address 21.0.2.2 nexthop-interface ge-0/0/1.0
!!!!!
--- lsping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
```

Traceroute can be used to further to verify the exact path traversed to R4 in flex-algo 133. In the next traceroute command to R4 with `200304` in the `label-stack` confirms that the path taken from R0 to R4 is R0->R2->R3-R4 and R0->R2->R5->R4.

Note that the per-prefix load balancing policies need to be configured to get the ECMP paths:

```
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept


regress@R0> traceroute mpls segment-routing spring-te label-stack labels 200304 nexthop-
address 21.0.2.2 nexthop-interface ge-0/0/1.0  egress 4.4.4.4

ttl        Label  Protocol   Address        Previous Hop   Probe Status
           1   200304  Static   21.0.2.2       (null)         Success
  FEC-Stack-Sent: SPRING-TE
  ttl      Label  Protocol   Address        Previous Hop   Probe Status
           2   200304  Static   21.2.3.2       21.0.2.2       Success
  FEC-Stack-Sent: SPRING-TE
  ttl      Label  Protocol  Address       Previous Hop   Probe Status
           3     3  Static    21.3.4.2       21.2.3.2       Egress
  FEC-Stack-Sent: SPRING-TE

  Path 1 via ge-0/0/1.0 destination 127.0.0.64

  ttl      Label  Protocol  Address       Previous Hop   Probe Status
           2   200304  Static   21.2.5.2       21.0.2.2       Success
  FEC-Stack-Sent: SPRING-TE
  ttl      Label  Protocol  Address       Previous Hop   Probe Status
           3     3  Static    21.4.5.1       21.2.5.2       Egress
  FEC-Stack-Sent: SPRING-TE

  Path 2 via ge-0/0/1.0 destination 127.0.0.65
```

Native flex-algo based MPLS ping and traceroute are also supported in future releases. This mechanism can be used to verify flex-algo paths when the intermediate routers support FEC validation. In Junos, this MPLS ping and traceroute uses the same FEC definition that is defined in RFC 8287 for prefix-SIDs:

```
regress@R0> ping mpls segment-routing isis 4.4.4.4 algorithm 133
!!!!!
--- lsping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss

regress@R0> traceroute mpls segment-routing isis 4.4.4.4 algorithm 133

ttl       Label  Protocol  Address       Previous Hop   Probe Status
          1   200304  ISIS 21.0.2.2      (null)         Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address       Previous Hop   Probe Status
          2   200304  ISIS 21.2.3.2      21.0.2.2       Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address       Previous Hop   Probe Status
          3     3  ISIS  21.3.4.2      21.2.3.2      Egress
  FEC-Stack-Sent: ISIS

  Path 1 via ge-0/0/1.0 destination 127.0.0.64

  ttl      Label  Protocol  Address       Previous Hop   Probe Status
          2   200304  ISIS 21.2.5.2      21.0.2.2       Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address       Previous Hop   Probe Status
          3     3  ISIS  21.4.5.1      21.2.5.2      Egress
  FEC-Stack-Sent: ISIS

  Path 2 via ge-0/0/1.0 destination 127.0.0.65
```

MPLS ping and traceroute mechanisms can be used either to manually validate the paths or for debugging purposes. Applications can be built using the output of traceroute to verify whether the expected path is being taken in the forwarding plane.

Protection for Flex-Algo Paths (TI_LFA _Flex-algo)

When post-convergence-LFA is enabled, the backup path for the routes is calculated based on flex-algo topology and metric-type. When the interface is enabled with post-convergence-LFA, the flex-algo routes will be programmed with TI-LFA backup paths corresponding to the flex-algo topology:

```
regress@R0> show route table inetcolor.0
…..
2.2.2.2-133<c>/64       [L-ISIS/14] 00:13:00, metric 1
                             >  to 21.0.2.2 via ge-0/0/1.0
                             to 21.0.3.2 via ge-0/0/2.0, Push 200102
….
```

Note that for 2.2.2.2 in flex-algo 133 topology, R0->R2 is the primary and backup is via R1->R3->R2. Note also that the TI-LFA path uses SIDs that have the N-bit set. The flex-algo SIDs advertised from each node should have the "node-segment" for TI-LFA computation to correctly pick-up flex-algo SIDs for building a backup path.

Migrating to Flex-Algo

All the nodes that participate in the flex-algo must be upgraded and configured to participate in the flex-algo. The routers that cannot be upgraded and do not support flex-algo will not participate in the flex-algo topology.

There are various ways in which service traffic can be mapped onto the flex-algo or TE paths. Color-based steering is one such mechanism that is supported in current releases. In future releases, CBF (CoS-based forwarding) and FBF (filter-based forwarding) will also be supported.

Color-based steering mechanisms use color extended community in the BGP service prefix to steer the traffic onto a particular flex-algo. On the egress node, a color extended community is configured and associated with a BGP service prefix.

Color assignment for BGP service prefixes

On R6 let's assign color 133 to the prefix 121.1.1.1:

```
set policy-options policy-statement bgp-export term one from route-filter 121.1.0.0/16 orlonger
set policy-options policy-statement bgp-export term one then community add com1

set policy-options community com1 members color:0:133
```

Color resolution should be enabled so that the BGP service prefix can resolve on the inetcolor.0 table. This can be achieved in two ways:

- `extended-nexthop-color`
- `resolution-map policy`

Using extended-nexthop-color

For BGP service family inet unicast and inet6 unicast, enabling `extended-nexthop-color` on the BGP session on both sides will enable the resolution on the inetcolor.0 table:

```
set protocols bgp group R6 family inet unicast extended-nexthop-color
```

Associate a color extended community with prefix 121.1.1.1 on R6.

Using resolution-map policy

For BGP service prefix belonging to inet VPN/inet6 VPN and all other unicast families, `resolution-map` policy needs to be configured and `resolution-mode` should be set to ip-color.

When the traffic for VPN prefixes need to use flex-algo paths, appropriate resolution-mode policies need to be configured.

In the lab setup, we enable inet-vpn family between the routers R0 and R7. We associate an interface to the VRF on R7 and advertise the associated interface address in the BGP inet-vpn family. We need to associate the color 133 in the color extended community and also configure resolution mode.

Normally, the VPN prefixes will resolve on inet.3 or inet.0. When the VPN prefixes have to resolve on the flex-algo, the VPN prefixes need to be associated with the color that maps to the flex-algo number or configured color in the flex-algo definition. The resolution needs to be performed on the inetcolor.0 table. The resolution of the protocol nexthop for the VPN prefix will be performed on the inetcolor table only when resolution mode is set. When the resolution should be performed on protocol <nexhop address: color>, the resolution mode should be set to ip-color. When the resolution needs to be performed on <0.0.0.0:color> then resolution-mode should be set to color-only.

Some network operators use anycast IP addresses for devices in the same region and build tunnels from the other regions only to the anycast IP addresses. The mesh needs to be built from region-to-region and not node-to-node. This significantly reduces the mesh of tunnels that need to be built. When this mechanism is used, the BGP protocol nexthop for services needs to use the anycast endpoint address, which may not be desirable as the legacy tunneling mechanisms are built on node loopbacks, and a fallback to legacy tunneling mechanism may be required.

Color-only resolution-mode enables color to be used as an IP address. In color-only mode, the BGP service prefix carries a color and that color is used to resolve in the tunnel table. For example, <0.0.0.0:100> is used as protocol next hop and the tunnels in the inetcolor table will also be installed as <0.0.0.0:100> by the controller.

Next we focus on ip-color resolution mode. Here is the configuration on R0 used to set the resolution-mode:

```
set protocols bgp group R7 import bgp_import_policy
set policy-options policy-statement bgp_import_policy term a from route-filter 99.1.1.0 orlonger
set policy-options policy-statement bgp_import_policy term a then community add red-color
set policy-options policy-statement bgp_import_policy term a then accept
set policy-options policy-statement bgp_import_policy term a then resolution-map rmap_1
set policy-options community red-color members color:0:133
set policy-options resolution-map rmap_1 mode ip-color
```

A BGP import policy is configured on R0 that adds a red color community to the VPN prefix and sets the resolution-map to rmap-1. In the `resolution-map rmap_1` the mode is set to `ip-color` mode. This ensures the VPN prefix resolves on inetcolor.0 with color 133:

```
regress@R0# run show route 99.1.1.0

vrf-red.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

99.1.1.0/24      *[BGP/170] 01:06:44, localpref 100, from 7.7.7.7
                   AS path: I, validation-state: unverified
                 >  to 21.0.2.2 via ge-0/0/1.0, Push 27, Push 200307(top)
```

Note that the VPN prefix is using the flex-algo label 200307 to send the traffic, which is flex-algo 133 SID for R7.

Shifting Traffic to Flex-Algo

When flex-algo is enabled, the flex-algo specific routes get created in the inetcolor.0 and mpls.0 tables. The traffic will start flowing on flex-algo routes when service prefix resolves on.

The color association with the prefixes will enable service prefixes to resolve on flex-algo routes as described in the section *Flex-Algo Migration*. The flex-algo SIDs can be associated with the same loopback on which the default ISIS node-SIDs are advertised. Since the service migration to flex-algo depends on color association on the service prefixes, separate loopback association will not be required. However, if an operator wants to assign a separate loopback IP address and associate flex-algo SID to that loopback, that would also work.

Let's look at the service prefix 121.1.1.1 on R0. We can see that the next hop has a label 2000306 which is a flex-algo 133 label for R6. also note that the best TE-metric path to 6.6.6.6 goes via interface ge-0/0/1.0 and the Route 121.1.1.1 resolves on this flex-algo route:

```
regress@R0# run show route 121.1.1.1

inet.0: 76 destinations, 107 routes (76 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

121.1.1.0/24       *[BGP/170] 00:00:39, localpref 100, from 6.6.6.6
                     AS path: I, validation-state: unverified
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200306


regress@R0# run ping 121.1.1.1 source 100.100.100.100 count 5
PING 121.1.1.1 (121.1.1.1): 56 data bytes
64 bytes from 121.1.1.1: icmp_seq=0 ttl=62 time=5.195 ms
64 bytes from 121.1.1.1: icmp_seq=1 ttl=62 time=115.911 ms
64 bytes from 121.1.1.1: icmp_seq=2 ttl=62 time=13.016 ms
64 bytes from 121.1.1.1: icmp_seq=3 ttl=62 time=25.337 ms
64 bytes from 121.1.1.1: icmp_seq=4 ttl=62 time=6.405 ms
--- 121.1.1.1 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 5.195/33.173/115.911/41.982 ms
```

Fallback Mechanisms

In certain use cases, when the flex-algo path is not available the operator might prefer to send the traffic on the best effort path instead of dropping the traffic. This can be achieved by copying the L-ISIS or the LDP routes into the inetcolor.0 using `ribgroups` copy.

In our lab we'll configure ribgroups for ISIS and copy the inet.3 routes into inetcolor.0. This ensures that when there is no flex-algo route for a prefix, it will take the best effort ISIS SPF path.

Configure a rib group to set the import RIB to copy routes from inet.3 to inetcolor.0:

```
set routing-options rib-groups isis_inet3to_inetcolor import-rib [ inet.3 inetcolor.0]
```

Set this `rib-group` in ISIS:

```
set protocols isis rib-group isis_inet3to_inetcolor
```

Verify that ISIS-SR routes identified by protocol L-ISIS are copied into the inetcolor.0 table:

```
regress@R0# run show route table inetcolor.0

inetcolor.0: 14 destinations, 14 routes (14 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1-0<c>/32
                   *[L-ISIS/14] 00:00:42, metric 2
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200101
                     to 21.0.3.2 via ge-0/0/2.0, Push 200101
```

```
1.1.1.1-133<c>/64
                  *[L-ISIS/14] 00:00:42, metric 12
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200301
2.2.2.2-0<c>/32
                  *[L-ISIS/14] 00:00:42, metric 1
                  >  to 21.0.2.2 via ge-0/0/1.0
2.2.2.2-133<c>/64
                  *[L-ISIS/14] 00:00:42, metric 1
                  >  to 21.0.2.2 via ge-0/0/1.0
3.3.3.3-0<c>/32
                  *[L-ISIS/14] 00:00:42, metric 1
                  >  to 21.0.3.2 via ge-0/0/2.0
                    to 21.0.2.2 via ge-0/0/1.0, Push 200103, Push 200101(top)
3.3.3.3-133<c>/64
                  *[L-ISIS/14] 00:00:42, metric 11
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200303
4.4.4.4-0<c>/32
                  *[L-ISIS/14] 00:00:42, metric 2
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200104
4.4.4.4-133<c>/64
                  *[L-ISIS/14] 00:00:42, metric 12
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200304
5.5.5.5-0<c>/32
                  *[L-ISIS/14] 00:00:42, metric 2
                  >  to 21.0.3.2 via ge-0/0/2.0, Push 200105
                    to 21.0.2.2 via ge-0/0/1.0, Push 200105, Push 200101(top)
5.5.5.5-133<c>/64
                  *[L-ISIS/14] 00:00:42, metric 2
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200305
6.6.6.6-0<c>/32
                  *[L-ISIS/14] 00:00:42, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200106
                    to 21.0.3.2 via ge-0/0/2.0, Push 200106
6.6.6.6-133<c>/64
                  *[L-ISIS/14] 00:00:42, metric 12
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200306
7.7.7.7-0<c>/32
                  *[L-ISIS/14] 00:00:42, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200107
                    to 21.0.3.2 via ge-0/0/2.0, Push 200107
7.7.7.7-133<c>/64
                  *[L-ISIS/14] 00:00:42, metric 3
                  >  to 21.0.2.2 via ge-0/0/1.0, Push 200307
```

Note that the L-ISIS routes are copied into the inetcolor.0 table with color being 0 and prefix length being 32. Setting a fallback mechanism ensures that when there is no route in one of the flex-algo topologies for a certain destination, the traffic follows the best effort ISIS path. Note that the example topology we have taken with flex-algo 133 configured on all nodes, the fallback mechanism may not offer any great benefit because when there are failures, the low latency flex-algo 133 will converge onto the next available lowest latency path. The fallback mechanisms are very useful when flex-algo is used to separate routing planes. If there is a failure in one plane and the plane is completely partitioned, the fallback mechanism will ensure traffic will be forwarded on the best effort path and not dropped.

Troubleshooting Flex-Algo

When something goes wrong, there should be enough information captured in displays to help troubleshoot and find the root cause. This is especially true when a new feature is introduced into the network. Let's discuss what can go wrong and how to determine the root cause in Junos.

Junos has the display command for each flex-algo as shown here:

```
egress@R0# run show isis spring flex-algorithm flex-algorithm-id 133
Flex Algo: 133
  Level: 2, Color: 133, Participating, FAD supported
           Winner: R0, Metric: 2, Calc: 0, Prio: 0, FAD supported
           Spf Version: 50
           Participation toggles: 1
           Last Route add: 0
           Last Route rem: 0
           Last Route chg: 0
           Last Route fail: 0
           Last Route def: 0
           Total Route add: 16
           Total Route rem: 0
           Total Route chg: 21
           Total Route fail: 0
           Total Route def: 0
           Topo refresh count: 2
           Full SPFs: 49, Partial SPFs: 1
 IS-IS Flex Algo 133 level 2 SPF log:
Start time       Elapsed (secs) Count Reason
Wed Nov 25 22:08:33    0.000460 1 Periodic SPF
Wed Nov 25 22:11:15    0.000250   38 Reconfig
Wed Nov 25 22:14:59    0.000453 1 Reconfig
Wed Nov 25 22:15:10    0.000270   20 Updated LSP R3.00-00
Wed Nov 25 22:15:15    0.000224   12 Updated LSP R3.00-00
Wed Nov 25 22:15:19    0.000307   11 Updated LSP R4.00-00
Wed Nov 25 22:15:24    0.000330 6 Updated LSP R6.00-00
Wed Nov 25 22:15:28    0.000215 3 Updated LSP R6.00-00
Wed Nov 25 22:16:55    0.000343 1 Reconfig
Wed Nov 25 22:19:35    0.000344 2 Reconfig
Wed Nov 25 22:20:05    0.000396 3 Reconfig
Wed Nov 25 22:25:53    0.000345 4 Reconfig
Wed Nov 25 22:26:04    0.000224 3 Updated LSP R1.00-00
Wed Nov 25 22:26:11    0.000337 3 Updated LSP R2.00-00
Wed Nov 25 22:26:17    0.000245 3 Updated LSP R3.00-00
Wed Nov 25 22:26:23    0.000309 2 Updated LSP R4.00-00
Wed Nov 25 22:26:27    0.000302 2 Updated LSP R5.00-00
Wed Nov 25 22:26:33    0.000316 3 Updated LSP R6.00-00
Wed Nov 25 22:26:36    0.000294 2 Updated LSP R7.00-00
Wed Nov 25 22:37:33    0.000501 2 Updated LSP R6.00-00
Wed Nov 25 22:47:08    0.000381 1 Reconfig
Wed Nov 25 22:47:53    0.000400 1 Reconfig
Wed Nov 25 23:01:18    0.000473 1 Periodic SPF
Wed Nov 25 23:15:46    0.000298 1 Periodic SPF
Wed Nov 25 23:29:04    0.000248 1 Periodic SPF
```

Okay, let's look at some possible problems and how to use the information in this display to identify these problems.

Misconfiguration of Flex-Algo Definition (FAD)

When FADSs are misconfigured on a few nodes, it can result in those nodes not participating in the flex-algo and that can result in forwarding paths that are not expected. In order to fix the problem, the operator needs to identify the misconfigured nodes in the network.

FAD is advertised by one or more nodes in the network. The nodes that receive the FAD advertisement from multiple nodes run a conflict resolution algorithm and determine a winning FAD. The conflict resolution algorithm involves comparing the FAD priority first and then the system-id of the FAD originator. If a node has a FAD configured, then it compares the FAD with its configured values. If there is a conflict in the FAD, the node removes itself from participating in the flex-algo.

Any misconfiguration in a FAD, on any node that wins the FAD conflict resolution, can cause other nodes to stop participating in the flex-algo. If the winning FAD has some constraints that are not supported by the node, it will also stop participating in that flex-algo. For example, if the implementation on R0 does not support the TE-metric metric type it will not participate in flex-algo 133.

In order to debug FAD misconfigurations, Junos provides a display command:

```
egress@R0# run show isis spring flex-algorithm flex-algorithm-id 133
Flex Algo: 133
  Level: 2, Color: 133, Participating, FAD supported
           Winner: R0, Metric: 2, Calc: 0, Prio: 0, FAD supported
           Spf Version: 50
        ……..
```

The display indicates that this node is participating in flex-algo 133. The FAD from R0 will be in force as that is the winner for the FAD. The display also has the metric type, calculation type, and priority that the winning FAD uses. This helps to identify whether the misconfiguration is on the winning FAD advertiser or the misconfiguration is on this node.

The `FAD supported` output indicates that the FAD that we received from R0, this node, supports all the constraints mentioned in the FAD and hence, is participating. If there is some constraint that this node does not support, then the node would not participate in the flex-algo and the reason for that would be recorded in this field.

SPF Trigger Problems

Networks often run into the problem of a CPU getting dominated on some nodes in the network. Repeated SPF computation can keep the CPU really busy and it is useful to identify reasons for the SPF trigger in order to resolve the problem.

Everything in IGP revolves around SPF computations. With flex-algo, IGP performs multiple SPF computations, once each for each flex-algo. It is very useful to have a display that shows how many times SPF was triggered and the reason for the SPF trigger for each flex-algo. Note that a change in a certain flex-algo topology will result in a SPF computation in the default topology and that particular topology. The flex-algo topologies that are not impacted by the change will not trigger SPF. This show command shows the SPF log for the flex-algo:

```
egress@R0# run show isis spring flex-algorithm flex-algorithm-id 133
…….

 IS-IS Flex Algo 133 level 2 SPF log:
Start time       Elapsed (secs) Count Reason
Wed Nov 25 22:08:33     0.000460 1 Periodic SPF
Wed Nov 25 22:11:15     0.000250   38 Reconfig
Wed Nov 25 22:14:59     0.000453 1 Reconfig
Wed Nov 25 22:15:10     0.000270   20 Updated LSP R3.00-00
Wed Nov 25 22:15:15     0.000224   12 Updated LSP R3.00-00
Wed Nov 25 22:15:19     0.000307   11 Updated LSP R4.00-00
Wed Nov 25 22:15:24     0.000330 6 Updated LSP R6.00-00
Wed Nov 25 22:15:28     0.000215 3 Updated LSP R6.00-00
Wed Nov 25 22:16:55     0.000343 1 Reconfig
Wed Nov 25 22:19:35     0.000344 2 Reconfig
Wed Nov 25 22:20:05     0.000396 3 Reconfig
Wed Nov 25 22:25:53     0.000345 4 Reconfig
Wed Nov 25 22:26:04     0.000224 3 Updated LSP R1.00-00
Wed Nov 25 22:26:11     0.000337 3 Updated LSP R2.00-00
Wed Nov 25 22:26:17     0.000245 3 Updated LSP R3.00-00
Wed Nov 25 22:26:23     0.000309 2 Updated LSP R4.00-00
Wed Nov 25 22:26:27     0.000302 2 Updated LSP R5.00-00
Wed Nov 25 22:26:33     0.000316 3 Updated LSP R6.00-00
Wed Nov 25 22:26:36     0.000294 2 Updated LSP R7.00-00
Wed Nov 25 22:37:33     0.000501 2 Updated LSP R6.00-00
Wed Nov 25 22:47:08     0.000381 1 Reconfig
Wed Nov 25 22:47:53     0.000400 1 Reconfig
Wed Nov 25 23:01:18     0.000473 1 Periodic SPF
Wed Nov 25 23:15:46     0.000298 1 Periodic SPF
Wed Nov 25 23:29:04     0.000248 1 Periodic SPF

…..
```

The above display shows the SPF history log for flex-algo 133. It records the time SPF was triggered and the reason for the trigger along with total time taken for the SPF computation. This information is very helpful in debugging any SPF related problems in a flex-algo.

Route Changes

Route changes in a network are another important parameter. It is useful to be able to have an insight into route changes that are taking place on a node. Continuously churning routes can cause unwanted load on the CPU. Junos has display information that records route changes and thus provides the ability to detect route churn on flex-algo routes:

```
egress@R0# run show isis spring flex-algorithm flex-algorithm-id 133
Flex Algo: 133
…….
           Spf Version: 50
          Participation toggles: 1
          Last Route add: 0
          Last Route rem: 0
          Last Route chg: 0
          Last Route fail: 0
          Last Route def: 0
          Total Route add: 16
          Total Route rem: 0
          Total Route chg: 21
          Total Route fail: 0
          Total Route def: 0
          Topo refresh count: 2
          Full SPFs: 49, Partial SPFs: 1
```

The SPF version specifies the number of times SPF was done on this topology. Participation toggles indicate if this node oscillated being part of flex-algo and not being part of flex-algo. The fields with "Last" tags indicate the route changes that occurred in last SPF run and Fields tagged with "Total" indicate the cumulative route changes due to all SPF runs. The "Topo refresh count" field indicates the number of times the flex-algo topology has been updated. This field is useful to debug if the flex-algo topology changes are being accounted for the SPF runs. `Full SPFs` and `Partial SPFs` indicate the number of SPFs since the inception of the flex-algo.

Internal ISIS Route Table Display

When ISIS computes the SPF and then computes the routes and next hops, it builds an internal routing table. ISIS then walks this internal routing table and downloads the changed routes into the RIB inet.0, inet6.0, etc. It is useful to be able to display this ISIS internal routing table in order to identify if there are problems in the route download to the RIB. A new option has been introduced to display flex-algo specific routes. The command and the output for flex-algo 133 is here:

```
regress@R0# run show isis route flex-algorithm-id 133
 IS-IS routing table          Current version: L1: 31 L2: 71
IPv4/IPv6 Routes
----------------
Prefix           L Version   Metric Type Interface   NH   Via             Backup Score
1.1.1.1/32       2   50     12 int  ge-0/0/1.0   IPV4 R2
2.2.2.2/32       2   50     1 int  ge-0/0/1.0  IPV4 R2
3.3.3.3/32       2   50     11 int  ge-0/0/1.0   IPV4 R2
4.4.4.4/32       2   50     12 int  ge-0/0/1.0   IPV4 R2
5.5.5.5/32       2   50     2 int  ge-0/0/1.0  IPV4 R2
6.6.6.6/32       2   50     13 int  ge-0/0/1.0   IPV4 R2
7.7.7.7/32       2   50     3 int  ge-0/0/1.0  IPV4 R2

MPLS Routes
-----------
```

```
Label            L Version   Metric Type Interface   NH   Via
200301/52        2   50     12 int  ge-0/0/1.0   MPLS   R2(2.2.2.2)
200302/52        2   50     1 int  ge-0/0/1.0  MPLS   Direct->R2(2.2.2.2)
200302/56        2   50     1 int  ge-0/0/1.0  MPLS   Direct->R2(2.2.2.2)
200303/52        2   50     11 int  ge-0/0/1.0   MPLS   R2(2.2.2.2)
                                       ge-0/0/2.0   MPLS   Direct->R3(3.3.3.3)
200303/56        2   50     11 int  ge-0/0/1.0   MPLS   R2(2.2.2.2)
                                       ge-0/0/2.0   MPLS   Direct->R3(3.3.3.3)
200304/52        2   50     12 int  ge-0/0/1.0   MPLS   R2(2.2.2.2)
200305/52        2   50     2 int  ge-0/0/1.0  MPLS   R2(2.2.2.2)
200306/52        2   50     13 int  ge-0/0/1.0   MPLS   R2(2.2.2.2)
200307/52        2   50     3 int  ge-0/0/1.0  MPLS   R2(2.2.2.2)
IPv4/IPv6->MPLS Routes
----------------------
Prefix           L Version   Metric Type Interface   NH   Via             Backup Score
1.1.1.1/32       2   50     12 int  ge-0/0/1.0   MPLS R2(2.2.2.2)
2.2.2.2/32       2   50     1 int  ge-0/0/1.0  MPLS Direct->R2(2.2.2.2)
3.3.3.3/32       2   50     11 int  ge-0/0/1.0   MPLS R2(2.2.2.2)
                                       ge-0/0/2.0   MPLS Direct->R3(3.3.3.3)
4.4.4.4/32       2   50     12 int  ge-0/0/1.0   MPLS R2(2.2.2.2)
5.5.5.5/32       2   50     2 int  ge-0/0/1.0  MPLS R2(2.2.2.2)
6.6.6.6/32       2   50     13 int  ge-0/0/1.0   MPLS R2(2.2.2.2)
7.7.7.7/32       2   50     3 int  ge-0/0/1.0  MPLS R2(2.2.2.2)
```

You can see that the output shows three different tables. The IPv4/IPv6 table shows the pure IP routes for the flex-algo 133. These routes are not downloaded to any RIB by default.

`MPLS Routes` shows the routes that get downloaded into mpls.0 tables for this flex-algo. `IPv4/IPv6 MPLS Routes` shows the labeled routes that get downloaded into inetcolor.0

Traffic Counters for Flex-algo SIDs

It is useful to be able to build a traffic matrix on a per flex-algo basis. Traffic that flows on a flex-algo does not get accounted for in default ISIS-SR counters. This is because flex-algo uses different labels in the forwarding plane. In this section we'll see how to allocate sensors for flex-algo labels and count the transit traffic on the flex-algo labels. Per-SID egress sensors provide the counters for IP and MPLS traffic on the ingress routers. Note that these counters are not supported for flex-algo routes at the time of writing this book, but will be supported in a future release.

The per-SID ingress counters are available for the flex-algo routes. These counters provide the transit traffic on a router for a particular flex-algo. To verify the counters for flex-algo SIDs enable the traffic statistics accounting. on R2:

```
set services analytics streaming-server jvision-server8 remote-address 100.1.1.2
set services analytics streaming-server jvision-server8 remote-port 2000
set services analytics export-profile export-common local-address 100.1.1.1
set services analytics export-profile export-common local-port 1000
set services analytics export-profile export-common reporting-rate 5
set services analytics export-profile export-common format gpb
set services analytics export-profile export-common transport udp
set services analytics sensor per-sid server-name jvision-server8
set services analytics sensor per-sid export-name export-common
set services analytics sensor per-sid polling-interval 5
set services analytics sensor per-sid resource /junos/services/segment-routing/sid/usage/
set services analytics sensor mpls server-name jvision-server8
set services analytics sensor mpls export-name export-common
set services analytics sensor mpls resource /junos/services/label-switched-path/usage/
set protocols isis source-packet-routing sensor-based-stats per-interface-per-member-link ingress
set protocols isis source-packet-routing sensor-based-stats per-interface-per-member-link egress
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
```

Verify the route to 121.1.1.1 is pointing to a flex-algo route:

```
inet.0: 78 destinations, 108 routes (78 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

121.1.1.0/24      *[BGP/170] 00:47:18, localpref 100, from 6.6.6.6
                    AS path: I, validation-state: unverified
                  > to 21.0.2.2 via ge-0/0/1.0, Push 200306
                    to 21.0.3.2 via ge-0/0/2.0, Push 200306

[edit]
regress@R0#
```

Send ping packets from R0 to 121.1.1.1:

```
regress@R0# run ping 121.1.1.1 source 100.100.100.100 count 10
PING 121.1.1.1 (121.1.1.1): 56 data bytes
64 bytes from 121.1.1.1: icmp_seq=0 ttl=62 time=16.952 ms
64 bytes from 121.1.1.1: icmp_seq=1 ttl=62 time=10.586 ms
64 bytes from 121.1.1.1: icmp_seq=2 ttl=62 time=5.800 ms
64 bytes from 121.1.1.1: icmp_seq=3 ttl=62 time=5.418 ms
64 bytes from 121.1.1.1: icmp_seq=4 ttl=62 time=5.677 ms
64 bytes from 121.1.1.1: icmp_seq=5 ttl=62 time=10.867 ms
64 bytes from 121.1.1.1: icmp_seq=6 ttl=62 time=44.580 ms
64 bytes from 121.1.1.1: icmp_seq=7 ttl=62 time=21.448 ms
64 bytes from 121.1.1.1: icmp_seq=8 ttl=62 time=10.987 ms
64 bytes from 121.1.1.1: icmp_seq=9 ttl=62 time=5.293 ms

--- 121.1.1.1 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 5.293/13.761/44.580/11.469 ms
```

On R2, 10 packets will be counted for the label 200306. This can be verified using PFE commands:

```
request pfe execute command "show agent sensor" target fpc0 | grep "per-sid" |no-more
3442950488  per-sid                 5000(0)

regress@R0>  request pfe execute command "show agent sensor id 3442950488" target fpc0 | no-more

SENT: Ukern command: show agent sensor id 3442950488

ID: 3442950488 Type: 2 Name: (per-sid)
```

```
Sensor Data: 209(181) bytes
Jvision Header: system: 10.220.15.251:100.1.1.1, slot: 0, time: Dec  2 06:42:24.863, sequence_
number: 35, sensor_name: per-sid:/junos/services/segment-routing/sid/usage/:/junos/services/
segment-routing/sid/usage/:PFE, version: 1.0
------------------------------------------------------------------------------------
          Packets        Bytes Packet RateByte Rate Inst  Counter-Name SID Name
--------------- ---------------- ----------- ------------ ---- ------------- --------
          10           880       0          0 0 oc-79  200306
------------------------------------------------------------------------------
Total 1 records.
------------------------------------------------------------------------------
   Export: Interval = 5000 msecs, Payload-size = 5000 bytes, Server = Src: 100.1.1.1:1000, Dst:
100.1.1.2:2000(VRF 0) , Qos = TOS:0x0, Hint:0x40000010, FC:0 PLP:0
   Filter: none
    Accounting:
          570 total successful reaps, 0 failed/aborted reap(s).
          1 reaps in latest reporting interval.
          1 packets in latest reporting interval.
          1 instances in latest reporting interval.
          36 total packets sent.
          570 wraps, 1 reap(s) on average to wrap.
    Last 10 wraps (UTC):
           2020 Dec 02 06:42:24:864
           2020 Dec 02 06:42:19:717
           2020 Dec 02 06:42:14:248
           2020 Dec 02 06:42:08:944
           2020 Dec 02 06:42:03:573
           2020 Dec 02 06:41:58:324
           2020 Dec 02 06:41:52:680
           2020 Dec 02 06:41:47:474
           2020 Dec 02 06:41:42:231
           2020 Dec 02 06:41:36:947
---------------------------
```

Chapter 10

Migrating LDP Over RSVP Transport to SR Over RSVP

This chapter's topology runs ISIS as the IGP, LDP at the edge, and RSVP in the core, all according to the following metrics:

- Routers R0, R1, R2, R3 are running LDP.
- Routers R2, R3, R4, R5 are 'super cores' and have RSVP enabled.
- Routers R4, R5, R6, R7 also have LDP enabled.
- RSVP LSPs are established between routers R2, R3, R4, and R5.

When many large networks have similar deployments, they run RSVP in the inner (super) core, LDP at the edge, and then tunnel LDP over the RSVP core (LDPoRS-VP). This solution reduces the number of full mesh RSVP LSPs.

MORE? For further reading on LDP tunneling see: https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/ldp-tunneling-edit-protocols-mpls.html and https://www.juniper.net/documentation/en_US/junos/topics/topic-map/ldp-configuration.html#id-24454.

Now let's migrate this network to SRoRSVP and remove LDP from the network. The network currently looks like Figure 10.1.

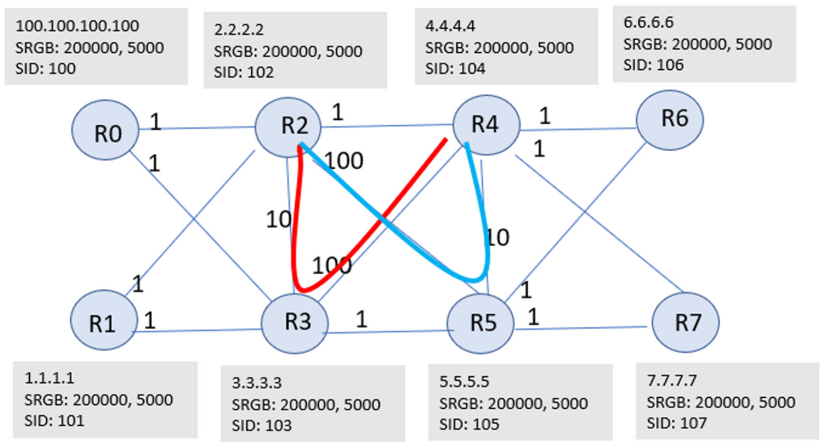


Figure 10.1 *Chapter 10's Topology*

There are two LSPs configured in the super core; the Red LSP and the Blue LSP:

- The Red RSVP LSP spans R2->R3->R4
- The Blue RSVP LSP spans R2->R5->R4

We will take the following steps to migrate our network:

- Verify the LSPs are up
- Enable ISIS-SR
- Verify the routes and traffic flows
- Enable and verify shortcuts for MPLS routes
- Verify path is correctly stitched to the RSVP
- Shift traffic to SR

Verify RSVP LSPs Are Up

Before starting any migration, you first need to ensure your network is working as expected, meaning in this case that the RSVP LSPs are up. Let's verify on R2, the initiator of both LSPs:

```
regress@R2# run show mpls lsp
Ingress LSP: 2 sessions
To            From          State Rt P   ActivePath       LSPname
4.4.4.4       2.2.2.2       Up     0 *   via_R5           blue_lsp
4.4.4.4       2.2.2.2       Up     0 *   via_R3           red_lsp
Total 2 displayed, Up 2, Down 0
```

```
Egress LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

As expected, both LSPs are up. The next step is to verify traffic initiated from, or routed via, R2 towards R6 is indeed using the RSVP tunnels. In order to do so let's take a look at the routing table:

```
regress@R2# run show route protocol ldp

inet.0: 69 destinations, 69 routes (69 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

224.0.0.2/32      *[LDP/9] 4d 19:52:38, metric 1
                    MultiRecv

inet.3: 19 destinations, 31 routes (19 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1/32        *[LDP/9] 00:13:42, metric 1
                  >  to 21.1.2.1 via ge-0/0/1.0
3.3.3.3/32        *[LDP/9] 00:13:15, metric 1
                  >  to 21.0.2.1 via ge-0/0/0.0, Push 19
                    to 21.1.2.1 via ge-0/0/1.0, Push 18
4.4.4.4/32        [LDP/9] 00:13:08, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
5.5.5.5/32        *[LDP/9] 00:12:49, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
6.6.6.6/32        *[LDP/9] 00:12:49, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
7.7.7.7/32        *[LDP/9] 00:12:48, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
100.100.100.100/32 *[LDP/9] 00:13:39, metric 1
                  >  to 21.0.2.1 via ge-0/0/0.0

iso.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

mpls.0: 35 destinations, 35 routes (35 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

19                *[LDP/9] 00:13:42, metric 1
                  >  to 21.1.2.1 via ge-0/0/1.0, Pop
19(S=0)           *[LDP/9] 00:13:42, metric 1
                  >  to 21.1.2.1 via ge-0/0/1.0, Pop
20                *[LDP/9] 00:13:39, metric 1
                  >  to 21.0.2.1 via ge-0/0/0.0, Pop
20(S=0)           *[LDP/9] 00:13:39, metric 1
                  >  to 21.0.2.1 via ge-0/0/0.0, Pop
21                *[LDP/9] 00:13:15, metric 1
                  >  to 21.0.2.1 via ge-0/0/0.0, Swap 19
                    to 21.1.2.1 via ge-0/0/1.0, Swap 18
22                *[LDP/9] 00:13:08, metric 1
```

```
                    >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                      to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
23                  *[LDP/9] 00:12:49, metric 1
                    >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                      to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
25                  *[LDP/9] 00:12:48, metric 1
                    >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                      to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
26                  *[LDP/9] 00:12:49, metric 1
                    >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                      to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
```

As you can observe from the output, you can reach R6 (6.6.6.6) via both LSPs:

```
6.6.6.6/32          *[LDP/9] 00:12:49, metric 1
                    >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                      to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
```

Enable ISIS-SR on All Routers and Restart Routing

Once the reachability and correct working of the network is ensured, it's time to enable ISIS-SR on all nodes. You do this by entering the following three lines of configuration on each router in the topology that will configure R0 and enable ISIS-SR:

```
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 4000
set protocols isis source-packet-routing node-segment ipv4-index 100
```

"`Set protocols isis source-packet-routing`" in itself enables SR for ISIS. Adding "`srgb`" configures the SRGB in ISIS-SR. The SRGB label range is based on the start label and the index range. The value of the start label indicates the start of the label range, and the value of the index range along with the value of the start label indicate the end of the label range:

```
start-label: start-label-value—Start of the SRGB label block.
Range: 16 through 1,048,575

index-range: index-range-value—Index range of the SRGB label block.
Range: 32 through 1,048,559

ipv4-index: index—IPv4 node segment index. The range is 0 through 199999.
```

Starting with Junos release 17.2, the maximum index for IPv4 node segment index is 199999. Ensure to increment the `ipv4-index` value per router as it needs to differ.

Configuring SRGB on a node requires MX and PTX devices to be running in enhanced services mode. When this mode is enabled, the router must be rebooted:

```
regress@R0>set chassis network-services enhanced-ip
```

NOTE Chapter 7 has details on configuring SRGBs and rebooting.

Verify L-ISIS Routes

After the router reboots you can verify if you are indeed advertising Label-Switched Paths into IS-IS. In order to verify `SPRING based MPLS` is enabled for ISIS issue the following command:

```
regress@R0> show isis overview
Instance: master
  Router ID: 100.100.100.100
  IPv6 Router ID: abcd::128:220:14:102
  Hostname: R0
  Sysid: 1282.2001.4102
  Areaid: 47.0005
  Adjacency holddown: enabled
  Maximum Areas: 3
  LSP life time: 1200
  Attached bit evaluation: enabled
  SPF delay: 200 msec, SPF holddown: 5000 msec, SPF rapid runs: 3
  IPv4 is enabled, IPv6 is enabled, SPRING based MPLS is enabled
  Traffic engineering: enabled
  Traffic engineering v6: disabled
  Restart: Disabled
          Helper mode: Enabled
  Layer2-map: Disabled
  flood-reduction: Disabled
  Source Packet Routing (SPRING): Enabled
          SRGB Config Range :
          SRGB Start-Label : 200000, SRGB Index-Range : 5000
          SRGB Block Allocation: Success
          SRGB Start Index : 200000, SRGB Size : 5000, Label-Range: [ 200000, 204999 ]
          Node Segments: Enabled
          Ipv4 Index : 100
          SRv6: Disabled
  Post Convergence Backup: Enabled
          Max labels: 3, Max spf: 100, Max Ecmp Backup: 1
  Level 1
          Internal route preference: 15
          External route preference: 160
          Prefix export count: 0
          Wide metrics are enabled, Narrow metrics are enabled
          Source Packet Routing is enabled
  Level 2
          Internal route preference: 18
          External route preference: 165
          Prefix export count: 0
          Wide metrics are enabled
          Source Packet Routing is enabled
```

Also, observing the routing table you can now see the following:

```
inet.3: 20 destinations, 33 routes (7 active, 0 holddown, 17 hidden)
+ = Active Route, - = Last Active, * = Both
1.1.1.1/32         [L-ISIS/14] 00:00:05, metric 1
                   >  to 21.1.2.1 via ge-0/0/1.0
                      to 21.0.2.1 via ge-0/0/0.0, Push 200101, Push 200103(top)
3.3.3.3/32         [L-ISIS/14] 00:00:05, metric 2
                   >  to 21.0.2.1 via ge-0/0/0.0, Push 200103
                      to 21.1.2.1 via ge-0/0/1.0, Push 200103
```

```
4.4.4.4/32         [L-ISIS/14] 00:16:08, metric 1
                   >  to 21.2.4.2 via ge-0/0/3.0
5.5.5.5/32         [L-ISIS/14] 00:00:05, metric 3
                   >  to 21.0.2.1 via ge-0/0/0.0, Push 200105
                      to 21.1.2.1 via ge-0/0/1.0, Push 200105
                      to 21.2.4.2 via ge-0/0/3.0, Push 200105
6.6.6.6/32         [L-ISIS/14] 00:03:15, metric 2
                   >  to 21.2.4.2 via ge-0/0/3.0, Push 200106
7.7.7.7/32         [L-ISIS/14] 00:03:12, metric 2
                   >  to 21.2.4.2 via ge-0/0/3.0, Push 200107
100.100.100.100/32  [L-ISIS/14] 00:00:05, metric 1
                   >  to 21.0.2.1 via ge-0/0/0.0
                      to 21.1.2.1 via ge-0/0/1.0, Push 200100, Push 200103(top)
```

Enable Shortcuts for MPLS Routes on R2

ISIS typically performs two independent computations. The first is performed without considering any LSPs and the result of the computation is stored in the inet.0 table. It's no different than traditional SPF computations and is always performed, even if an IGP shortcut is disabled.

The second computation is performed only on LSPs as a logical interface. Each LSP's egress router is considered. The list of destinations whose shortest path traverses the egress router (established during the first computation) is placed in the inet.3 routing table. These destinations are given to the egress router of the LSP as a next hop.

Interior gateway protocol (IGP) shortcuts, also called traffic-engineering shortcuts, provide a tool by which the link-state IGP (OSPF or IS-IS) in an AS can consider an LSP in its shortest-path-first (SPF) calculations.

When you use `traffic-engineering bgp` (which is the default) and IGP shortcuts, the traffic engineering solution is only used for BGP AS-external route resolution. However, traffic to AS-internal destinations can also be mapped to LSPs. To accomplish this, `traffic-engineering bgp-igp` is enabled. Thus, RSVP installs the MPLS prefixes into the inet.0 table rather than the inet.3 table. As a result, the MPLS LSPs are installed in the forwarding table.

This approach provides a practical application whenever heavy traffic is routed to specific destinations within an AS, such as server farms.

An important point about IGP shortcuts, whether used alone or in conjunction with `traffic-engineering BGP-IGP`, is that IGP adjacencies are never formed across the LSPs. The IGP sees the LSP as a single data link, but does not view the egress router as a potential peer and does not forward hello messages across the LSP. Also, RSVP messages are never forwarded over LSPs, preventing the possibility of an LSP being inadvertently built within another LSP.

MORE? Further reading on ISIS traffic engineering shortcuts can be found in the following resources:

- https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/traffic-engineering-edit-protocols-isis.html
- https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/shortcuts-edit-protocols-isis.html
- https://www.juniper.net/documentation/en_US/junos/topics/example/isis-traffic-engineering.html

Shortcuts are enabled on a per-node basis. You do not need to coordinate with other nodes. Let's enable some shortcuts for labeled ISIS by issuing:

```
set protocols isis traffic-engineering family inet-mpls shortcuts
```

Now check that the shortcuts are active:

```
regress@R2# run show route protocol isis table inet.3

inet.3: 20 destinations, 33 routes (7 active, 0 holddown, 17 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1/32         [L-ISIS/14] 00:00:08, metric 1
                   >  to 21.1.2.1 via ge-0/0/1.0
                     to 21.0.2.1 via ge-0/0/0.0, Push 200101, Push 200103(top)
3.3.3.3/32         [L-ISIS/14] 00:00:08, metric 2
                   >  to 21.0.2.1 via ge-0/0/0.0, Push 200103
                     to 21.1.2.1 via ge-0/0/1.0, Push 200103
4.4.4.4/32         [L-ISIS/14] 00:00:08, metric 1
                   >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
5.5.5.5/32         [L-ISIS/14] 00:00:08, metric 3
                   >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
6.6.6.6/32         [L-ISIS/14] 00:00:08, metric 2
                   >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
7.7.7.7/32         [L-ISIS/14] 00:00:08, metric 2
                   >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
100.100.100.100/32   [L-ISIS/14] 00:00:08, metric 1
                   >  to 21.0.2.1 via ge-0/0/0.0
                     to 21.1.2.1 via ge-0/0/1.0, Push 200100, Push 200103(top)

regress@R2# run show route 6.6.6.6

inet.0: 73 destinations, 73 routes (73 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

6.6.6.6/32         *[IS-IS/18] 00:00:54, metric 2
                   >  to 21.2.4.2 via ge-0/0/3.0

inet.3: 20 destinations, 33 routes (7 active, 0 holddown, 17 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
6.6.6.6/32        *[LDP/9] 00:00:54, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [L-ISIS/14] 00:00:54, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
```

Notice that the next hops of the L-ISIS routes are changed to the LSPs. This is in line with what's expected.

TIP Shortcuts can be enabled for different address families in the ISIS configuration: `set protocols isis traffic-engineering` **`family inet`** `shortcuts` will shortcut ISIS IP routes; `set protocols isis traffic-engineering` **`family inet-mpls`** `shortcuts` will shortcut only L-ISIS SID routes.

Next, enable shortcuts on the R2 router for ISIS IP routes:

```
set protocols isis traffic-engineering family inet shortcuts
```

Again, verify the ISIS routes on R2. Note that ISIS routes are now stitched to RSVP:

```
regress@R2# run show route table inet.3

inet.3: 20 destinations, 33 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1/32        *[LDP/9] 00:00:08, metric 1
                  >  to 21.1.2.1 via ge-0/0/1.0
                  [L-ISIS/14] 00:00:09, metric 1
                  >  to 21.1.2.1 via ge-0/0/1.0
                     to 21.0.2.1 via ge-0/0/0.0, Push 200101, Push 200103(top)
3.3.3.3/32        *[LDP/9] 00:00:09, metric 1
                  >  to 21.0.2.1 via ge-0/0/0.0, Push 19
                     to 21.1.2.1 via ge-0/0/1.0, Push 18
                  [L-ISIS/14] 00:00:09, metric 2
                  >  to 21.0.2.1 via ge-0/0/0.0, Push 200103
                     to 21.1.2.1 via ge-0/0/1.0, Push 200103
4.4.4.4/32        *[RSVP/7/1] 00:00:09, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp

                  [LDP/9] 00:00:08, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [L-ISIS/14] 00:00:08, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [IS-IS/18] 00:00:08, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
5.5.5.5/32        *[LDP/9] 00:00:08, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
```

```
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [L-ISIS/14] 00:00:08, metric 3
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [IS-IS/18] 00:00:08, metric 3
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
6.6.6.6/32        *[LDP/9] 00:00:08, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [L-ISIS/14] 00:00:08, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [IS-IS/18] 00:00:08, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
7.7.7.7/32        *[LDP/9] 00:00:08, metric 1
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [L-ISIS/14] 00:00:08, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
                  [IS-IS/18] 00:00:08, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
21.3.4.0/24       *[IS-IS/18] 00:00:08, metric 101
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
21.4.5.0/24       *[IS-IS/18] 00:00:08, metric 11
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
21.4.6.0/24       *[IS-IS/18] 00:00:08, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
21.4.7.0/24       *[IS-IS/18] 00:00:08, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
21.5.6.0/24       *[IS-IS/18] 00:00:08, metric 3
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
21.5.7.0/24       *[IS-IS/18] 00:00:08, metric 3
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
22.5.6.0/24       *[IS-IS/18] 00:00:08, metric 12
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
66.66.66.66/32    *[IS-IS/18] 00:00:08, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
77.77.77.77/32    *[IS-IS/18] 00:00:08, metric 2
                  >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                    to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
100.100.100.100/32 *[LDP/9] 00:00:08, metric 1
                  >  to 21.0.2.1 via ge-0/0/0.0
                  [L-ISIS/14] 00:00:09, metric 1
                  >  to 21.0.2.1 via ge-0/0/0.0
                    to 21.1.2.1 via ge-0/0/1.0, Push 200100, Push 200103(top)
128.220.14.122/32  *[IS-IS/18] 00:00:08, metric 2
                    to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
```

```
                   >  to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
128.220.15.77/32    *[IS-IS/18] 00:00:08, metric 2
                     to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                   >  to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
128.220.15.81/32    *[IS-IS/18] 00:00:08, metric 3
                   >  to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                     to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
128.220.16.149/32   *[IS-IS/18] 00:00:08, metric 1
                     to 21.2.3.2 via ge-0/0/2.0, label-switched-path red_lsp
                   >  to 21.2.5.2 via ge-0/0/4.0, label-switched-path blue_lsp
```

NOTE The remaining part of this chapter assumes both *family inet shortcuts* and *family inet-mpls shortcuts* are enabled.

Verify the Path is Correctly Stitched to RSVP

You can verify the path is correctly stitched to RSVP by issuing `mpls ping` and `traceroute` commands.

MPLS Ping

Use `mpls ping` functionality to diagnose the state of label-switched paths (LSPs), Layer 2 and Layer 3 VPNs, and Layer 2 circuits. You can ping an MPLS endpoint using various options. You can send variations of ICMP echo request packets to the specified MPLS endpoint.

When you use the ping MPLS task from Junos operating as the inbound (ingress) node at the entry point of an LSP or VPN, the routing platform sends probe packets into the LSP or VPN. Depending on how the LSP or VPN outbound (egress) node at the remote endpoint of the connection replies to the probes, you can determine the connectivity of the LSP or VPN.

Each probe is an echo request sent to the LSP or VPN exit point as an MPLS packet with a UDP payload. If the outbound node receives the echo request, it checks the contents of the probe and returns a value in the UDP payload of the response packet. If the initiator receives the response packet, it reports a successful ping response. Responses that take longer than two seconds are identified as failed probes.

MORE? Further good reading on `mpls ping` at the Juniper TechLibrary: https://www.juniper.net/documentation/en_US/junos-space-apps/connectivity-services-director2.0/topics/concept/ping-mpls-for-csd.html

Okay. Let's issue `mpls ping` to check if the destination is reachable:

```
regress@R0> ping mpls segment-routing isis 6.6.6.6
!!!!!
--- lsping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
```

Traceroute with MPLS segment-routing ISIS

Next you can use `traceroute` to a remote host for a SR label-switched path added by the ISIS protocol. Use `traceroute mpls segment-routing isis` as a debugging tool to locate MPLS label-switched path forwarding issues in a network.

MORE? Further reading on `traceroute mpls segment-routing isis` can be found in the TechLibrary: https://www.juniper.net/documentation/en_US/junos/topics/reference/command-summary/traceroute-mpls-isis.html.

```
regress@R0> traceroute mpls segment-routing isis 6.6.6.6
  Probe options: ttl 64, retries 3, wait 10, paths 16, exp 7, fanout 16

  ttl      Label  Protocol  Address        Previous Hop   Probe Status
         1   200106  ISIS     21.0.2.2       (null)         Success
  FEC-Stack-Sent: ISIS FEC-Change-Recieved: PUSH-RSVP
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
         2   39  RSVP-TE  21.2.3.2       21.0.2.2       Success
  FEC-Stack-Sent: RSVP,ISIS
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
         3     3  RSVP-TE   21.3.4.2       21.2.3.2       Egress
  FEC-Stack-Sent: RSVP,ISIS FEC-Change-Recieved: POP-RSVP
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
         3     3  RSVP-TE   21.3.4.2       21.2.3.2       Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
         4     3  ISIS      21.4.6.2       21.3.4.2       Egress
  FEC-Stack-Sent: ISIS

  Path 1 via ge-0/0/1.0 destination 127.0.0.64

  ttl      Label  Protocol  Address        Previous Hop   Probe Status
         1   200106  ISIS     21.0.3.2       (null)         Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
         2   200106  ISIS     21.3.5.2       21.0.3.2       Success
  FEC-Stack-Sent: ISIS
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
         3     3  ISIS      21.5.6.2       21.3.5.2       Egress
  FEC-Stack-Sent: ISIS
```

Note that path 2 via ge-0/0/2.0, destination 127.0.1.64, the traffic is load-balanced across red_lsp and blue_lsp on R2. The paths for 6.6.6.6 for LDP and L-ISIS match with each other.

Ping MPLS LDP

Check the operability of MPLS LDP-signaled label-switched path (LSPs) connections by typing Ctrl+c to interrupt a `ping mpls` command:

```
regress@R0> ping mpls ldp 6.6.6.6
!!!!!
--- lsping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
```

MORE? Further reading on `ping mpls ldp`: https://www.juniper.net/documentation/en_US/junos/topics/reference/command-summary/ping-mpls-ldp.html .

Traceroute MPLS LDP

Traceroute to a remote host for an MPLS label-switched path signaled by the LDP. Use traceroute MPLS LDP as a debugging tool to locate MPLS label-switched path forwarding issues in a network.

MORE? For further reading on `traceroute mpls ldp` visit: https://www.juniper.net/documentation/en_US/junos/topics/reference/command-summary/traceroute-mpls-ldp.html .

```
regress@R0> traceroute mpls ldp 6.6.6.6
  Probe options: ttl 64, retries 3, wait 10, paths 16, exp 7, fanout 16

  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           1   34  LDP      21.0.3.2       (null)         Success
  FEC-Stack-Sent: LDP
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           2   21  LDP      21.3.5.2       21.0.3.2       Success
  FEC-Stack-Sent: LDP
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           3      3  LDP      21.5.6.2       21.3.5.2       Egress
  FEC-Stack-Sent: LDP

  Path 1 via ge-0/0/2.0 destination 127.0.0.64

  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           1   72  LDP      21.0.2.2       (null)         Success
  FEC-Stack-Sent: LDP FEC-Change-Recieved: PUSH-RSVP
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           2   39  RSVP-TE  21.2.3.2       21.0.2.2       Success
  FEC-Stack-Sent: RSVP,LDP
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           3      3  RSVP-TE   21.3.4.2       21.2.3.2       Egress
  FEC-Stack-Sent: RSVP,LDP FEC-Change-Recieved: POP-RSVP
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           3      3  RSVP-TE   21.3.4.2       21.2.3.2       Success
  FEC-Stack-Sent: LDP
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           4      3  LDP      21.4.6.2       21.3.4.2       Egress
  FEC-Stack-Sent: LDP
```

Note that path 2 via ge-0/0/1.0 is destination 127.0.1.64.

Shift Traffic to ISIS-SR

After enabling ISIS-SR, installing the shortcuts, and verifying the adjusted routing, you are now ready to start shifting traffic to ISIS-SR. In order to do so you need to make sure the labeled ISIS SR routes have a better preference then LDP. In order to do that issue the following command:

```
regress@R0# set protocols isis level 2 labeled-preference 7
```

NOTE Always make sure you fully understand the impact on the whole routing ecosystem when changing, lowering, or increasing the preference of routes as this potentially influences other routes as well. Make sure to thoroughly test this upfront in your lab *before* making these changes in the production network.

After changing the preference let's see how 6.6.6.6 now resolves in inet.3:

```
[edit]
regress@R0# run show route 6.6.6.6

inet.0: 66 destinations, 70 routes (66 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

6.6.6.6/32         *[IS-IS/18] 2d 11:45:14, metric 3
                   >  to 21.0.2.2 via ge-0/0/1.0
                      to 21.0.3.2 via ge-0/0/2.0

inet.3: 7 destinations, 14 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

6.6.6.6/32         *[L-ISIS/7] 00:00:07, metric 3
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 200106
                      to 21.0.3.2 via ge-0/0/2.0, Push 200106
                    [LDP/9] 00:15:27, metric 1
                   >  to 21.0.2.2 via ge-0/0/1.0, Push 72
                      to 21.0.3.2 via ge-0/0/2.0, Push 34

[edit]
```

As expected, the route to 6.6.6.6 is now preferred over ISIS-SR.

Chapter 11

Migrating LDP Transport to LDP over SR-TE

In some scenarios it isn't possible to upgrade parts of the network from LDP to SR. Segment Routing Traffic Engineering (SR-TE) offers options to transport LDP over it: (https://www.juniper.net/documentation/en_US/junos/topics/concept/tunneling-ldp-over-srte.html).

This chapter looks at such a scenario where R0, R1, R6, and R7 can't be upgraded and so we'll enable SR on the core routers (R2, R3, R4, and R5) only and then transport LDP over it.

Tunneling LDP over SR-TE is beneficial because it:

- Enables seamless migration of LDP over SR-TE in the core network.
- Provides flexible connectivity options to accommodate multiple topologies, protocols, and domains.
- Enables interoperability between LDP and SR capable devices.
- Leverages SR-TE load sharing capabilities.
- Provides faster restoration of network connectivity using TI-LFA within the SR-TE domain. SR using TI-LFA routes the traffic instantly to a backup or to an alternate path if the primary path fails or becomes unavailable.

Tunneling LDP over SR-TE Overview

In a typical service provider network, it is common to use MPLS signaling transport protocols such as LDP at the edge and core routers. As a service provider, you might be looking at gradually migrating or deploying SR-TE in the core network, eliminating the need for MPLS signaling protocols such as LDP.

In this case there are some routers (R0, R1, R6, and R7) in the network that do not support SR capabilities and we want to continue using those routers (running LDP) without a need for an upgrade. In such scenarios, the LDP over SR-TE tunneling feature provides the interoperability benefit of using the routers that are not SR capable (running LDP) with routers that are SR capable (running SR-TE).

The LDP LSPs are tunneled through the SR-TE network, enabling interworking of LDP LSPs with SR-TE LSPs. For example, if you have LDP domains on the provider edge network and SR-TE in the core network, then you can connect the LDP domains over SR-TE, as shown in Figure 11.1. You can continue to have IGP (OSPF or IS-IS) in the traffic engineered core and in the surrounding LDP domains.

Tunneling LDP over SR-TE provides consistency that coexists with both LDP LSPs and SR-TE LSPs.

You can also tunnel LDP over SR-TE between LDP domains connected to inter-region core networks. For example, if you have multiple regional LDP domains connected to the regional SR-TE core networks you can tunnel LDP across the inter-region SR-TE core networks.

Enabling Segment Routing In the Core

The network is running ISIS and LDP on all routers. As stated previously, we assume R0, R1, R6, and R7 are not SR capable so those will continue to run LDP. The core routers R2, R3, R4, and R5 are SR capable, so we will enable SR on those:

```
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 4000
set protocols isis source-packet-routing node-segment ipv4-index 100
```

Next verify the routes:

```
regress@R2# run show route table inet.3 protocol isis

inet.3: 7 destinations, 10 routes (7 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

4.4.4.4/32         [L-ISIS/14] 18:38:19, metric 1
                   >  to 21.2.4.2 via ge-0/0/3.0
5.5.5.5/32         [L-ISIS/14] 18:37:40, metric 3
                   >  to 21.2.4.2 via ge-0/0/3.0, Push 200105
```

Note that there is no L-ISIS route to 3.3.3.3. This is because the next hop is going towards R0 and R1, which are *not* SR capable. You can observe that the LDP routes are created for 3.3.3.3:

```
regress@R2# run show route 3.3.3.3

inet.0: 73 destinations, 73 routes (73 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
3.3.3.3/32         *[IS-IS/18] 18:32:05, metric 2
                   >  to 21.0.2.1 via ge-0/0/0.0
                     to 21.1.2.1 via ge-0/0/1.0

inet.3: 7 destinations, 10 routes (7 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

3.3.3.3/32         *[LDP/9] 18:32:05, metric 1
                   >  to 21.0.2.1 via ge-0/0/0.0, Push 25
                     to 21.1.2.1 via ge-0/0/1.0, Push 23
```

Using `mpls ping` and `traceroute` you can verify that the SR paths are installed correctly. Review how to do this in the previous chapters.

Configuring SR-TE LSPs

Look at Figure 11.1 and assume link R2-R4 is congested and as a result traffic needs to be shifted away from that link. This is where TE comes in handy.

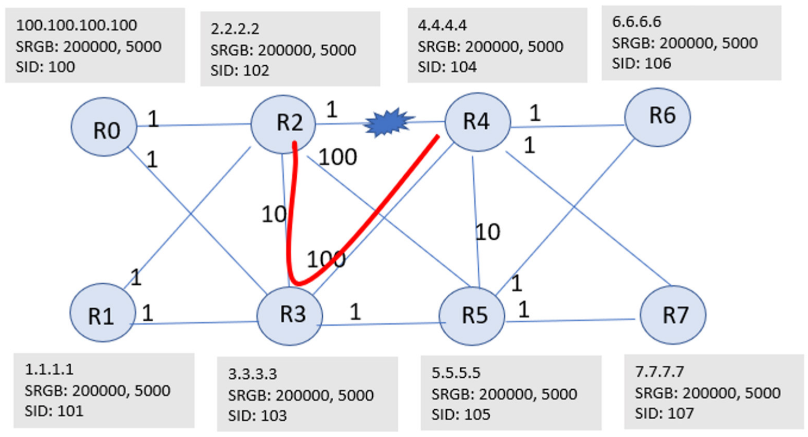


Fig 11.1 Red SR-TE LSP Created Along R2->R3->R4

Now configure the SR-TE LSP taking the path R2->R3->R4 using static adjacency SIDs (adj-SIDs). The advantage of using static adj-SID is that it will remain stable and does not change when a link flap occurs.

We define static adj-SIDs on R2 and R3 so that we can build a path from R2->R3->R4. Static adj-SIDs can be protected or unprotected. While advertising protected static adjacency SIDs, make sure protection is enabled on the link.

Configure the LSP on R2

```
set protocols isis interface ge-0/0/2.0 level 2 post-convergence-lfa
set protocols isis backup-spf-options use-post-convergence-lfa
set protocols isis interface ge-0/0/2.0 level 2 ipv4-adjacency-segment protected label 900001
set protocols mpls label-range static-label-range 900000 904000
```

Configure the LSP on R3

```
set protocols isis interface ge-0/0/4.0 level 2 post-convergence-lfa
set protocols isis backup-spf-options use-post-convergence-lfa
set protocols isis interface ge-0/0/4.0 level 2 ipv4-adjacency-segment protected label 900001

set protocols mpls label-range static-label-range 900000 904000
```

Verify the LSP

Now that the LSP is configured you should see that reflected in the routing table:

```
regress@R2# run show route label 900001

mpls.0: 30 destinations, 30 routes (30 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

900001            *[L-ISIS/14] 00:03:12, metric 0
                  >  to 21.2.3.2 via ge-0/0/2.0, Pop
900001(S=0)       *[L-ISIS/14] 00:03:12, metric 0
                  >  to 21.2.3.2 via ge-0/0/2.0, Pop


regress@R2# run show configuration protocols source-packet-routing |display set
set protocols source-packet-routing segment-list sr_via_R3 hop1 label 900001
set protocols source-packet-routing segment-list sr_via_R3 hop2 label 900001
set protocols source-packet-routing source-routing-path sr_red_lsp to 4.4.4.4
set protocols source-packet-routing source-routing-path sr_red_lsp primary sr_via_R3
```

NOTE These examples show the configuration using static SR-TE but the same could be done with Juniper Paragon Pathfinder (formerly NorthStar Controller) with PCEP. See this link for further details: https://www.juniper.net/us/en/products-services/network-automation/paragon-pathfinder/.

```
regress@R2# run show route 4.4.4.4

inet.0: 73 destinations, 73 routes (73 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

4.4.4.4/32        *[IS-IS/18] 19:23:23, metric 1
                  >  to 21.2.4.2 via ge-0/0/3.0

inet.3: 7 destinations, 10 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
4.4.4.4/32       *[SPRING-TE/8] 00:00:22, metric 1, metric2 1
                 >  to 21.2.3.2 via ge-0/0/2.0, Push 900001
                 [LDP/9] 19:23:19, metric 1
                 >  to 21.2.4.2 via ge-0/0/3.0
                 [L-ISIS/14] 19:23:23, metric 1
                 >  to 21.2.4.2 via ge-0/0/3.0


regress@R2> show spring-traffic-engineering lsp
To            State  LSPname
4.4.4.4       Up     sr_red_lsp


Total displayed LSPs: 1 (Up: 1, Down: 0)
```

Verify SR-TE LSP Using MPLS Ping and Traceroute

As shown in the earlier chapters `mpls ping` and traceroute are very useful tools to verify reachability:

```
regress@R2> ping mpls segment-routing spring-te source-routing-path sr_red_lsp
!!!!!
--- lsping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss


regress@R2> traceroute mpls segment-routing spring-te source-routing-path sr_red_lsp skip-fec-
validation
  Probe options: retries 3, exp 7

  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           1   900001  Static    21.2.3.2       (null)         Unhelpful
  FEC-Stack-Sent: SPRING-TE
  ttl      Label  Protocol  Address        Previous Hop   Probe Status
           2                    4.4.4.4        21.2.3.2       Egress
  FEC-Stack-Sent: SPRING-TE

  Path 1 via ge-0/0/2.0 destination 127.0.0.64
```

Enable LDP Over SR-TE (LDPoSR-TE)

The final step in this chapter is to tunnel LDP over SR-TE. In order to do so we need to enable `ldp tunneling` and again install the shortcuts that we discussed earlier.

NOTE Keep in mind that preference is important here: SR-TE and RSVP can be configured with relative preferences. The tunnel source that has the highest preference will be chosen to shortcut and when not configured the next preferred source will be used.

Enable ldp_tunneling on SR-TE LSP

```
regress@R2# set protocols source-packet-routing source-routing-path sr_red_lsp ldp-tunneling

regress@R2# run show configuration protocols isis traffic-engineering |display set
set protocols isis traffic-engineering tunnel-source-protocol spring-te
set protocols isis traffic-engineering family inet shortcuts
```

Note that the currently used `tunnel-source-protocol` (`spring-te`) is supported for family inet only. If family `inet-mpls` is also enabled when tunnel source is `spring-te`, then the shortcuts will not be created as normal spring-te paths use the node-SIDs and there is the possibility of loop when SR labels shortcut onto SR-TE.

Verify the shortcuts are in place

```
regress@R2# run show route 4.4.4.4

inet.0: 73 destinations, 73 routes (73 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

4.4.4.4/32       *[IS-IS/18] 00:00:08, metric 1
                 >  to 21.2.4.2 via ge-0/0/3.0

inet.3: 20 destinations, 27 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

4.4.4.4/32       *[SPRING-TE/8] 00:00:08, metric 1, metric2 1
                 >  to 21.2.3.2 via ge-0/0/2.0, Push 900001
                 [LDP/9] 00:00:08, metric 1
                 >  to 21.2.3.2 via ge-0/0/2.0, Push 900001
                 [L-ISIS/14] 00:00:08, metric 1
                 >  to 21.2.4.2 via ge-0/0/3.0
                 [IS-IS/18] 00:00:08, metric 1
                 >  to 21.2.3.2 via ge-0/0/2.0, Push 900001
```

You can see that the LDP and the ISIS routes are now stitched onto the spring-te tunnel. In other words, you have successfully enabled LDPoSR-TE.

Verify End-to-End Reachability

The next and last step is to verify from R0, in the LDP domain, reachability to R6. Let’s do so by issuing the `mpls ping` command:

```
regress@R0> ping mpls ldp 6.6.6.6
!!!!!
--- lsping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
```

Appendices

Appendix A: Baseline Lab Configuration

Here are the baseline configurations used in the reference topology per chapter.

Legacy Network Configuration

R0 Configuration:

```
regress@R0> show configuration interfaces |display set

set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 description ConnectedToR2
set interfaces ge-0/0/1 unit 0 vlan-id 2
set interfaces ge-0/0/1 unit 0 family inet address 21.0.2.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 description ConnectedToR3
set interfaces ge-0/0/2 unit 0 vlan-id 3
set interfaces ge-0/0/2 unit 0 family inet address 21.0.3.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls

set interfaces lo0 unit 0 family inet address 100.100.100.100/32
set interfaces lo0 unit 0 family iso address 47.0005.0019.2168.1100.00

regress@R0> show configuration protocols|display set
regress@R0> show configuration protocols |display set
set protocols bgp group R7 type internal
set protocols bgp group R7 local-address 100.100.100.100
set protocols bgp group R7 family inet unicast
set protocols bgp group R7 neighbor 7.7.7.7
set protocols bgp group R6 type internal
set protocols bgp group R6 local-address 100.100.100.100
set protocols bgp group R6 family inet unicast
set protocols bgp group R6 neighbor 6.6.6.6

set protocols bgp group R1 type internal
set protocols bgp group R1 local-address 100.100.100.100
set protocols bgp group R1 family inet unicast
set protocols bgp group R1 neighbor 1.1.1.1

set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 point-to-point
set protocols ospf area 0.0.0.0 interface lo0.0 passive

regress@R0> show configuration routing-options |display set
set routing-options router-id 100.100.100.100
set routing-options autonomous-system 60030
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept
```

R1 configuration:

```
regress@R1> show configuration interfaces |display set
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 description ConnectedToR2
set interfaces ge-0/0/1 unit 0 vlan-id 12
set interfaces ge-0/0/1 unit 0 family inet address 21.1.2.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 description ConnectedToR3
set interfaces ge-0/0/2 unit 0 vlan-id 13
set interfaces ge-0/0/2 unit 0 family inet address 21.1.3.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.1/32
set interfaces lo0 unit 0 family iso address 47.0005.0019.2168.1101.00


regress@R1> show configuration protocols |display set
set protocols bgp group R7 type internal
set protocols bgp group R7 local-address 1.1.1.1
set protocols bgp group R7 family inet unicast
set protocols bgp group R7 neighbor 7.7.7.7
set protocols bgp group R6 type internal
set protocols bgp group R6 local-address 1.1.1.1
set protocols bgp group R6 family inet unicast
set protocols bgp group R0 type internal
set protocols bgp group R0 local-address 1.1.1.1
set protocols bgp group R0 family inet unicast
set protocols bgp group R0 neighbor 100.100.100.100
set protocols bgp group R6 neighbor 6.6.6.6
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 point-to-point
set protocols ospf area 0.0.0.0 interface lo0.0 passive

regress@R1> show configuration routing-options |display set
set routing-options router-id 1.1.1.1
set routing-options autonomous-system 60030
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept
```

R2 Configuration:

```
regress@R2> show configuration interfaces |display set
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 0 description ConnectedToR0
set interfaces ge-0/0/0 unit 0 vlan-id 2
set interfaces ge-0/0/0 unit 0 family inet address 21.0.2.2/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 description ConnectedToR1
set interfaces ge-0/0/1 unit 0 vlan-id 12
set interfaces ge-0/0/1 unit 0 family inet address 21.1.2.2/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 description ConnectedToR3
set interfaces ge-0/0/2 unit 0 vlan-id 23
set interfaces ge-0/0/2 unit 0 family inet address 21.2.3.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 description ConnectedToR4
set interfaces ge-0/0/3 unit 0 vlan-id 24
set interfaces ge-0/0/3 unit 0 family inet address 21.2.4.1/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 description ConnectedToR5
set interfaces ge-0/0/4 unit 0 vlan-id 25
set interfaces ge-0/0/4 unit 0 family inet address 21.2.5.1/24
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls

set interfaces lo0 unit 0 family inet address 2.2.2.2/32
set interfaces lo0 unit 0 family iso address 47.0005.0019.2168.1102.00

regress@R2> show configuration protocols |display set
```

```
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 metric 100
set protocols ospf area 0.0.0.0 interface lo0.0 passive

regress@R2> show configuration routing-options |display set
set routing-options router-id 2.2.2.2
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept
```

R3 Configuration:

```
regress@R3> show configuration interfaces |display set
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 0 description ConnectedToR0
set interfaces ge-0/0/0 unit 0 vlan-id 3
set interfaces ge-0/0/0 unit 0 family inet address 21.0.3.2/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls

set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 description ConnectedToR1
set interfaces ge-0/0/2 unit 0 vlan-id 13
set interfaces ge-0/0/2 unit 0 family inet address 21.1.3.2/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 description ConnectedToR2
set interfaces ge-0/0/3 unit 0 vlan-id 23
set interfaces ge-0/0/3 unit 0 family inet address 21.2.3.2/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 description ConnectedToR4
set interfaces ge-0/0/4 unit 0 vlan-id 34
set interfaces ge-0/0/4 unit 0 family inet address 21.3.4.1/24
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 description ConnectedToR5
set interfaces ge-0/0/5 unit 0 vlan-id 35
set interfaces ge-0/0/5 unit 0 family inet address 21.3.5.1/24
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces ge-0/0/6 vlan-tagging
set interfaces ge-0/0/8 vlan-tagging
set interfaces ge-0/0/8 unit 0 description ConnectedToR7
set interfaces ge-0/0/8 unit 0 vlan-id 37
set interfaces ge-0/0/8 unit 0 family inet address 21.3.7.1/24
set interfaces ge-0/0/8 unit 0 family iso
set interfaces ge-0/0/8 unit 0 family mpls
deactivate interfaces ge-0/0/8
set interfaces lo0 unit 0 family inet address 3.3.3.3/32
set interfaces lo0 unit 0 family iso address 47.0005.0019.2168.1103.00

regress@R3> show configuration protocols |display set

set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 metric 100
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 point-to-point
set protocols ospf area 0.0.0.0 interface lo0.0 passive

regress@R3> show configuration routing-options |display set
set routing-options router-id 3.3.3.3
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept
```

R4 Configuration:

```
regress@R4> show configuration interfaces |display set
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 description ConnectedToR4
set interfaces ge-0/0/2 unit 0 vlan-id 24
set interfaces ge-0/0/2 unit 0 family inet address 21.2.4.2/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 description ConnectedToR4
set interfaces ge-0/0/3 unit 0 vlan-id 34
set interfaces ge-0/0/3 unit 0 family inet address 21.3.4.2/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 description ConnectedToR5
set interfaces ge-0/0/4 unit 0 vlan-id 45
set interfaces ge-0/0/4 unit 0 family inet address 21.4.5.1/24
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 description ConnectedToR6
set interfaces ge-0/0/5 unit 0 vlan-id 46
set interfaces ge-0/0/5 unit 0 family inet address 21.4.6.1/24
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces ge-0/0/6 vlan-tagging
set interfaces ge-0/0/6 unit 0 description ConnectedToR7
set interfaces ge-0/0/6 unit 0 vlan-id 47
set interfaces ge-0/0/6 unit 0 family inet address 21.4.7.1/24
set interfaces ge-0/0/6 unit 0 family iso
set interfaces ge-0/0/6 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 4.4.4.4/32
set interfaces lo0 unit 0 family iso address 47.0005.0019.2168.1104.00

regress@R4> show configuration protocols |display set

set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 metric 100
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/6.0 point-to-point
set protocols ospf area 0.0.0.0 interface lo0.0 passive
regress@R4> show configuration routing-options |display set
set routing-options router-id 4.4.4.4
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept
```

R5 Configuration:

```
regress@R5> show configuration interfaces |display set

set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 description ConnectedToR2
set interfaces ge-0/0/2 unit 0 vlan-id 25
set interfaces ge-0/0/2 unit 0 family inet address 21.2.5.2/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 description ConnectedToR3
set interfaces ge-0/0/3 unit 0 vlan-id 35
set interfaces ge-0/0/3 unit 0 family inet address 21.3.5.2/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 description ConnectedToR4
set interfaces ge-0/0/4 unit 0 vlan-id 45
set interfaces ge-0/0/4 unit 0 family inet address 21.4.5.2/24
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 description ConnectedToR6
set interfaces ge-0/0/5 unit 0 vlan-id 56
set interfaces ge-0/0/5 unit 0 family inet address 21.5.6.1/24
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces ge-0/0/6 vlan-tagging
set interfaces ge-0/0/6 unit 0 description ConnectedToR7
set interfaces ge-0/0/6 unit 0 vlan-id 57
set interfaces ge-0/0/6 unit 0 family inet address 21.5.7.1/24
set interfaces ge-0/0/6 unit 0 family iso
set interfaces ge-0/0/6 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 5.5.5.5/32
set interfaces lo0 unit 0 family iso address 47.0005.0019.2168.1105.00

regress@R5> show configuration protocols |display set

set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 metric 100
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/6.0 point-to-point
set protocols ospf area 0.0.0.0 interface lo0.0 passive

regress@R5> show configuration routing-options |display set
set routing-options router-id 5.5.5.5
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept
```

R6 Configuration:

```
regress@R6> show configuration interfaces |display set
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 0 description ConnectedToR0
set interfaces ge-0/0/0 unit 0 vlan-id 6
set interfaces ge-0/0/0 unit 0 family inet address 21.0.6.2/24
set interfaces ge-0/0/0 unit 0 family inet address 121.1.1.1/24
set interfaces ge-0/0/0 unit 0 family inet address 121.2.1.1/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls

set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 description ConnectedToR4
set interfaces ge-0/0/5 unit 0 vlan-id 46
set interfaces ge-0/0/5 unit 0 family inet address 21.4.6.2/24
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces ge-0/0/6 vlan-tagging
set interfaces ge-0/0/6 unit 0 description ConnectedToR5
set interfaces ge-0/0/6 unit 0 vlan-id 56
set interfaces ge-0/0/6 unit 0 family inet address 21.5.6.2/24
set interfaces ge-0/0/6 unit 0 family iso
set interfaces ge-0/0/6 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 6.6.6.6/32
set interfaces lo0 unit 0 family iso address 47.0005.0019.2168.1106.00

regress@R6> show configuration protocols |display set
set routing-options autonomous-system 60030
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept



set protocols bgp group R0 type internal
set protocols bgp group R0 local-address 6.6.6.6
set protocols bgp group R0 family inet unicast
set protocols bgp group R0 neighbor 100.100.100.100
set protocols bgp group R0 export bgp_export_policy

set protocols bgp group R1 type internal
set protocols bgp group R1 local-address 6.6.6.6
set protocols bgp group R1 family inet unicast
set protocols bgp group R1 neighbor 1.1.1.1
set protocols bgp group R1 export bgp_export_policy

set protocols bgp group R1 neighbor 1.1.1.1
set protocols bgp group R7 type internal
set protocols bgp group R7 local-address 6.6.6.6
set protocols bgp group R7 family inet unicast
set protocols bgp group R7 neighbor 7.7.7.7

set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/6.0 point-to-point
set protocols ospf area 0.0.0.0 interface lo0.0 passive

set policy-options policy-statement bgp_export_policy term one from route-
filter 121.1.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term one then next-hop 6.6.6.6
set policy-options policy-statement bgp_export_policy term one then accept

set policy-options policy-statement bgp_export_policy term two from route-
filter 121.2.0.0/16  orlonger
set policy-options policy-statement bgp_export_policy term two then next-hop 6.6.6.6
set policy-options policy-statement bgp_export_policy term two then accept
```

```
regress@R6> show configuration routing-options |display set
set routing-options router-id 6.6.6.6
set routing-options autonomous-system 60030
set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept
```

R7 Configuration:

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 0 description ConnectedToR0
set interfaces ge-0/0/0 unit 0 vlan-id 7
set interfaces ge-0/0/0 unit 0 family inet address 21.0.7.2/24
set interfaces ge-0/0/0 unit 0 family inet address 131.1.1.1/24
set interfaces ge-0/0/0 unit 0 family inet address 131.2.1.1/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 description ConnectedToR4
set interfaces ge-0/0/5 unit 0 vlan-id 47
set interfaces ge-0/0/5 unit 0 family inet address 21.4.7.2/24
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces ge-0/0/6 vlan-tagging
set interfaces ge-0/0/6 unit 0 description ConnectedToR5
set interfaces ge-0/0/6 unit 0 vlan-id 57
set interfaces ge-0/0/6 unit 0 family inet address 21.5.7.2/24
set interfaces ge-0/0/6 unit 0 family iso
set interfaces ge-0/0/6 unit 0 family mpls

set interfaces lo0 unit 0 family inet address 7.7.7.7/32
set interfaces lo0 unit 0 family iso address 47.0005.0019.2168.1107.00

regress@R7> show configuration protocols |display set
set protocols bgp group R0 type internal
set protocols bgp group R0 local-address 7.7.7.7
set protocols bgp group R0 family inet unicast
set protocols bgp group R0 export bgp_export_policy
set protocols bgp group R0 neighbor 100.100.100.100
set protocols bgp group R1 type internal
set protocols bgp group R1 local-address 7.7.7.7
set protocols bgp group R1 family inet unicast
set protocols bgp group R1 export bgp_export_policy
set protocols bgp group R1 neighbor 1.1.1.1
set protocols bgp group R6 type internal
set protocols bgp group R6 local-address 7.7.7.7
set protocols bgp group R6 family inet unicast
set protocols bgp group R6 neighbor 6.6.6.6


set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 point-to-point
set protocols ospf area 0.0.0.0 interface ge-0/0/6.0 point-to-point
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

```
regress@R7> show configuration routing-options |display set
set routing-options router-id 7.7.7.7
set routing-options autonomous-system 60030

regress@R7# run show configuration policy-options |display set
set policy-options policy-statement bgp_export_polcy term two then next-hop 7.7.7.7
set policy-options policy-statement bgp_export_policy term one from route-
filter 131.1.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term one then next-hop 7.7.7.7
set policy-options policy-statement bgp_export_policy term one then accept
set policy-options policy-statement bgp_export_policy term two from route-
filter 131.2.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term two then accept

set routing-options forwarding-table export pplb
set policy-options policy-statement pplb term t1 then load-balance per-packet
set policy-options policy-statement pplb term t1 then accept
```

Enabling OSPF-SR

Router R0 Configuration:

```
set chassis network-services enhanced-ip
set protocols ospf source-packet-routing srgb start-label 400000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing node-segment ipv4-index 100
```

Router R1 Configuration:

```
set chassis network-services enhanced-ip
set protocols ospf source-packet-routing srgb start-label 400000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing node-segment ipv4-index 101
```

Router R2 Configuration:

```
set chassis network-services enhanced-ip
set protocols ospf source-packet-routing srgb start-label 400000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing node-segment ipv4-index 102
```

Router R3 Configuration:

```
set chassis network-services enhanced-ip
set protocols ospf source-packet-routing srgb start-label 400000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing node-segment ipv4-index 103
```

Router R4 Configuration:

```
set chassis network-services enhanced-ip
set protocols ospf source-packet-routing srgb start-label 400000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing node-segment ipv4-index 104
```

Router R5 Configuration:

```
set chassis network-services enhanced-ip
set protocols ospf source-packet-routing srgb start-label 400000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing node-segment ipv4-index 105
```

Router R6 Configuration:

```
set chassis network-services enhanced-ip
set protocols ospf source-packet-routing srgb start-label 400000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing node-segment ipv4-index 106
```

Router R7 Configuration:

```
set chassis network-services enhanced-ip
set protocols ospf source-packet-routing srgb start-label 400000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing node-segment ipv4-index 107
```

Enabling TI-LFA protection

Router R0 Configuration:

```
set protocols ospf area 0 interface ge-0/0/1.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/2.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

Router R1 Configuration:

```
set protocols ospf area 0 interface ge-0/0/1.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/2.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

Router R2 Configuration:

```
set protocols ospf area 0 interface ge-0/0/0.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/1.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/2.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/3.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/4.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

Router R3 Configuration:

```
set protocols ospf area 0 interface ge-0/0/0.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/2.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/3.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/4.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/5.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

Router R4 Configuration:

```
set protocols ospf area 0 interface ge-0/0/2.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/3.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/4.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/5.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/6.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

Router R5 Configuration:

```
set protocols ospf area 0 interface ge-0/0/2.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/3.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/4.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/5.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/6.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

Router R6 Configuration:

```
set protocols ospf area 0 interface ge-0/0/5.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/6.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

Router R7 Configuration:

```
set protocols ospf area 0 interface ge-0/0/5.0 post-convergence-lfa
set protocols ospf area 0 interface ge-0/0/6.0 post-convergence-lfa
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

Shifting Traffic to OSPF-SR

Router R6 Configuration:

```
set interfaces lo0.0 family inet address 66.66.66.66/32
set policy-options policy-statement sid_assign_policy term one from route-filter 66.66.66.66/32 exact
set policy-options policy-statement sid_assign_policy term one then prefix-segment index 606
set policy-options policy-statement sid_assign_policy term one then accept
set protocols ospf source-packet-routing prefix-segment sid_assign_policy

set policy-options policy-statement bgp_export_policy term one from route-
filter 121.1.0.0/16 orlonger

set policy-options policy-statement bgp_export_policy term one then next-hop 66.66.66.66
set policy-options policy-statement bgp_export_policy term one then accept
set policy-options policy-statement bgp_export_policy term two from route-
filter 121.2.0.0/16  orlonger

set policy-options policy-statement bgp_export_policy term two then next-hop 6.6.6.6
set policy-options policy-statement bgp_export_policy term two then accept
commit
```

Router R7 Configuration:

```
set policy-options policy-statement bgp_export_policy term one from route-
filter 131.1.0.0/16 orlonger

set policy-options policy-statement bgp_export_policy term one then next-hop 77.77.77.77
set policy-options policy-statement bgp_export_policy term one then accept

set interfaces lo0.0 family inet address 77.77.77.77/32
set policy-options policy-statement sid_assign_policy term one from route-filter 77.77.77.77/32 exact
set policy-options policy-statement sid_assign_policy term one then prefix-segment index 707
set policy-options policy-statement sid_assign_policy term one then accept
set protocols ospf source-packet-routing prefix-segment sid_assign_policy
```

Changing OSPF-SR Route Preference on All Routers

```
set protocols ospf labeled-preference 8
```

Enabling ISIS-SR and TI-LFA Protection

R0 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 100
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 post-convergence-lfa
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 post-convergence-lfa

set protocols isis interface ge-0/0/1.0 level 2 metric 1
set protocols isis interface ge-0/0/2.0 level 2 metric 1

set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
```

R1 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 101
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 post-convergence-lfa
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable

set protocols isis interface ge-0/0/1.0 level 2 metric 1
set protocols isis interface ge-0/0/2.0 level 2 metric 1
```

R2 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 102
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/0.0 level 2 post-convergence-lfa
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 post-convergence-lfa
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable

set protocols isis interface ge-0/0/2.0 level 2 metric 10
set protocols isis interface ge-0/0/4.0 level 2 metric 100

set protocols isis interface ge-0/0/0.0 level 2 metric 1
set protocols isis interface ge-0/0/1.0 level 2 metric 1
set protocols isis interface ge-0/0/3.0 level 2 metric 1
```

R3 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 103
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 2 metric 10
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface ge-0/0/4.0 level 2 metric 100
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable

set protocols isis interface ge-0/0/3.0 level 2 metric 10
set protocols isis interface ge-0/0/4.0 level 2 metric 100

set protocols isis interface ge-0/0/0.0 level 2 metric 1
set protocols isis interface ge-0/0/2.0 level 2 metric 1
set protocols isis interface ge-0/0/5.0 level 2 metric 1
```

R4 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 104
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 2 metric 100
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface ge-0/0/4.0 level 2 metric 10
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable

set protocols isis interface ge-0/0/3.0 level 2 metric 100
set protocols isis interface ge-0/0/4.0 level 2 metric 10

set protocols isis interface ge-0/0/2.0 level 2 metric 1

set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 level 2 metric 1
```

R5 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 105
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 metric 100
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface ge-0/0/4.0 level 2 metric 10
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable

set protocols isis interface ge-0/0/4.0 level 2 metric 10
set protocols isis interface ge-0/0/2.0 level 2 metric 100

set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 level 2 metric 1
set protocols isis interface ge-0/0/3.0 level 2 metric 1
```

R6 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 106
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/6.0 point-to-point

set protocols isis interface ge-0/0/0.0 level 1 disable
set protocols isis interface ge-0/0/0.0 level 2 disable

set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 level 2 metric 1
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
```

R7 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 107
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/6.0 point-to-point

set protocols isis interface ge-0/0/0.0 level 1 disable
set protocols isis interface ge-0/0/0.0 level 2 disable

set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 level 2 metric 1

set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
```

Enabling Traffic Accounting

R0 Configurations:

```
set services analytics streaming-server jvision-server8 remote-address 100.1.1.2
set services analytics streaming-server jvision-server8 remote-port 2000
set services analytics export-profile export-common local-address 100.1.1.1
set services analytics export-profile export-common local-port 1000
set services analytics export-profile export-common reporting-rate 5
set services analytics export-profile export-common format gpb
set services analytics export-profile export-common transport udp
set services analytics sensor per-sid server-name jvision-server8
set services analytics sensor per-sid export-name export-common
set services analytics sensor per-sid polling-interval 5
set services analytics sensor per-sid resource /junos/services/segment-routing/sid/usage/
set services analytics sensor mpls server-name jvision-server8
set services analytics sensor mpls export-name export-common
set services analytics sensor mpls resource /junos/services/label-switched-path/usage/


set protocols isis source-packet-routing sensor-based-stats per-interface-per-member-link ingress
set protocols isis source-packet-routing sensor-based-stats per-interface-per-member-link egress
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
```

Flex-Algo Configurations

R0 Configuration:

```
set routing-options flex-algorithm 133 definition metric-type te-metric
set routing-options flex-algorithm 133 definition spf

set protocols isis source-packet-routing flex-algorithm 133
set protocols isis export flex_sid_policy
set protocols isis traffic-engineering advertisement always

set protocols isis interface ge-0/0/1.0 level 2 te-metric 1
set protocols isis interface ge-0/0/2.0 level 2 te-metric 100


set policy-options policy-statement flex_sid_policy term one from route-
filter 100.100.100.100/32 exact
set policy-options policy-statement flex_sid_policy term one then prefix-
segment algorithm 133 index 300 node-segment
```

R1 Configuration:

```
set protocols isis source-packet-routing flex-algorithm 133
set protocols isis export flex_sid_policy
set protocols isis traffic-engineering advertisement always

set protocols isis interface ge-0/0/1.0 level 2 te-metric 100
set protocols isis interface ge-0/0/2.0 level 2 te-metric 1
set policy-options policy-statement flex_sid_policy term one from route-filter 1.1.1.1/32 exact
set policy-options policy-statement flex_sid_policy term one then prefix-
segment algorithm 133 index 301 node-segment
```

R2 Configuration:

```
set protocols isis source-packet-routing flex-algorithm 133
set protocols isis export flex_sid_policy
set protocols isis traffic-engineering advertisement always

set protocols isis interface ge-0/0/0.0 level 2 te-metric 1
set protocols isis interface ge-0/0/1.0 level 2 te-metric 100
set protocols isis interface ge-0/0/2.0 level 2 te-metric 10
set protocols isis interface ge-0/0/3.0 level 2 te-metric 100
set protocols isis interface ge-0/0/4.0 level 2 te-metric 1
```

R3 Configuration:

```
set protocols isis source-packet-routing flex-algorithm 133
set protocols isis export flex_sid_policy
set protocols isis traffic-engineering advertisement always

set protocols isis interface ge-0/0/0.0 level 2 te-metric 100
set protocols isis interface ge-0/0/2.0 level 2 te-metric 1
set protocols isis interface ge-0/0/3.0 level 2 te-metric 10
set protocols isis interface ge-0/0/4.0 level 2 te-metric 1
set protocols isis interface ge-0/0/5.0 level 2 te-metric 100

set policy-options policy-statement flex_sid_policy term one from route-filter 3.3.3.3/32 exact
set policy-options policy-statement flex_sid_policy term one then prefix-
segment algorithm 133 index 303 node-segment
```

R4 Configuration:

```
set protocols isis source-packet-routing flex-algorithm 133
set protocols isis export flex_sid_policy
set protocols isis traffic-engineering advertisement always

set protocols isis interface ge-0/0/2.0 level 2 te-metric 1
set protocols isis interface ge-0/0/3.0 level 2 te-metric 100
set protocols isis interface ge-0/0/4.0 level 2 te-metric 100
set protocols isis interface ge-0/0/5.0 level 2 te-metric 1
set protocols isis interface ge-0/0/6.0 level 2 te-metric 10

set policy-options policy-statement flex_sid_policy term one from route-filter 4.4.4.4/32 exact
set policy-options policy-statement flex_sid_policy term one then prefix-
segment algorithm 133 index 304 node-segment
```

R5 Configuration:

```
set protocols isis source-packet-routing flex-algorithm 133
set protocols isis export flex_sid_policy
set protocols isis traffic-engineering advertisement always
set protocols isis traffic-engineering advertisement always
set protocols isis interface ge-0/0/2.0 level 2 te-metric 100
set protocols isis interface ge-0/0/3.0 level 2 te-metric 1
set protocols isis interface ge-0/0/4.0 level 2 te-metric 1
set protocols isis interface ge-0/0/5.0 level 2 te-metric 100
set protocols isis interface ge-0/0/6.0 level 2 te-metric 10

set policy-options policy-statement flex_sid_policy term one from route-filter 5.5.5.5/32 exact
set policy-options policy-statement flex_sid_policy term one then prefix-
segment algorithm 133 index 305 node-segment
```

R6 Configuration:

```
set protocols bgp group R6 family inet unicast extended-nexthop-color

set protocols isis source-packet-routing flex-algorithm 133
set protocols isis export flex_sid_policy
set protocols isis traffic-engineering advertisement always

set protocols isis interface ge-0/0/5.0 level 2 te-metric 100
set protocols isis interface ge-0/0/6.0 level 2 te-metric 1

set policy-options policy-statement flex_sid_policy term one from route-filter 6.6.6.6/32 exact
set policy-options policy-statement flex_sid_policy term one then prefix-
segment algorithm 133 index 306 node-segment

set policy-options policy-statement bgp_export_policy term one from route-
filter 121.1.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term one then community add com1

set policy-options community com1 members color:0:133
```

R7 Configuration:

```
set protocols isis source-packet-routing flex-algorithm 133
set protocols isis export flex_sid_policy
set protocols isis traffic-engineering advertisement always

set protocols isis interface ge-0/0/5.0 level 2 te-metric 1
set protocols isis interface ge-0/0/6.0 level 2 te-metric 100

set policy-options policy-statement flex_sid_policy term one from route-filter 7.7.7.7/32 exact
set policy-options policy-statement flex_sid_policy term one then prefix-
segment algorithm 133 index 307 node-segment

Service mapping for VPN prefixes
-------------------------------------------------
```

Configurations on R0:

```
set routing-instances vrf-red instance-type vrf
set routing-instances vrf-red route-distinguisher 10.1.1.1:1
set routing-instances vrf-red vrf-import VPN-A-import
set routing-instances vrf-red vrf-export VPN-A-export

set policy-options policy-statement VPN-A-export term a from family inet
set policy-options policy-statement VPN-A-export term a from protocol direct
set policy-options policy-statement VPN-A-export term a then community add VPN-A
set policy-options policy-statement VPN-A-export term a then accept
set policy-options policy-statement VPN-A-import term a from protocol bgp
set policy-options policy-statement VPN-A-import term a from community VPN-A
set policy-options policy-statement VPN-A-import term a then community add red-color
set policy-options policy-statement VPN-A-import term a then accept
set policy-options policy-statement VPN-A-import term b then reject
set policy-options policy-statement bgp-import term a then community add red-color
set policy-options policy-statement bgp-import term a then accept
set policy-options policy-statement bgp-import term a then resolution-map rmap-1

set policy-options community VPN-A members target:65000:1
set policy-options community red-color members color:0:133
set policy-options resolution-map rmap-1 mode ip-color

 set protocols bgp group R7 import bgp-import
set protocols bgp group R7 family inet-vpn unicast

activate interfaces ge-0/0/1
set interface ge-0/0/1.0 family inet address 99.1.1.1/24
```

Configurations on R7:

```
set routing-instances vrf-red instance-type vrf
set routing-instances vrf-red interface ge-0/0/1.0
set routing-instances vrf-red route-distinguisher 10.1.1.1:1
set routing-instances vrf-red vrf-import VPN-A-import
set routing-instances vrf-red vrf-export VPN-A-export
set routing-instances vrf-red vrf-target target:100:1
deactivate routing-instances vrf-red vrf-target
set routing-instances vrf-red vrf-table-label


set policy-options policy-statement VPN-A-export term a then community add VPN-A
set policy-options policy-statement VPN-A-export term a then accept
set policy-options policy-statement VPN-A-import term a from protocol bgp
set policy-options policy-statement VPN-A-import term a from community VPN-A
set policy-options policy-statement VPN-A-import term a then community add red-color
set policy-options policy-statement VPN-A-import term a then accept
set policy-options policy-statement VPN-A-import term a then resolution-map rmap-1
set policy-options policy-statement VPN-A-import term b then reject

set policy-options policy-statement bgp-import term a then community add red-color
set policy-options policy-statement bgp-import term a then accept
set policy-options policy-statement bgp-import term a then resolution-map rmap-1


set policy-options community VPN-A members target:65000:1
set policy-options community com1 members color:0:133
set policy-options community green-color members color:0:90
set policy-options community red-color members color:0:133
set policy-options resolution-map rmap-1 mode ip-color

set protocols bgp group R0 family inet-vpn unicast

commit
```

Baseline LDPoRSVP Configuration

Undo all the OSPF and ISIS configurations done in previous chapters and keep only the interface address configuration. Start with basic ISIS configuration shown here.

R0 Configuration:

```
regress@R0> show configuration protocols isis|display set

set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 metric 1

set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 metric 1

set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0

regress@R0> show configuration protocols ldp |display set
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

set protocols mpls interface all
set protocols mpls interface fxp0.0 disable

protocols bgp group R7 type internal
set protocols bgp group R7 local-address 100.100.100.100
set protocols bgp group R7 family inet unicast
set protocols bgp group R7 neighbor 7.7.7.7
set protocols bgp group R6 type internal
set protocols bgp group R6 local-address 100.100.100.100
set protocols bgp grouset p R6 family inet unicast
set protocols bgp group R6 neighbor 6.6.6.6

set protocols bgp group R1 type internal
set protocols bgp group R1 local-address 100.100.100.100
set protocols bgp group R1 family inet unicast
set protocols bgp group R1 neighbor 1.1.1.1

set routing-options router-id 100.100.100.100
set routing-options autonomous-system 60030
```

R1 Configuration:

```
regress@R1> show configuration protocols isis |display set
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 metric 1
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 metric 1
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0

regress@R1> show configuration protocols ldp |display set
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

set protocols mpls interface all
set protocols mpls interface fxp0.0 disable

set protocols bgp group R7 type internal
set protocols bgp group R7 local-address 1.1.1.1
set protocols bgp group R7 family inet unicast
set protocols bgp group R7 neighbor 7.7.7.7
set protocols bgp group R6 type internal
set protocols bgp group R6 local-address 1.1.1.1
set protocols bgp group R6 family inet unicast
set protocols bgp group R0 type internal
set protocols bgp group R0 local-address 1.1.1.1
set protocols bgp group R0 family inet unicast
set protocols bgp group R0 neighbor 100.100.100.100
set protocols bgp group R6 neighbor 6.6.6.6

set routing-options router-id 1.1.1.1
set routing-options autonomous-system 60030
```

R2 Configuration:

```
regress@R2> show configuration protocols isis|display set
set protocols isis level 2 wide-metrics-only
set protocols isis traffic-engineering family inet shortcuts
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/0.0 level 2 metric 1

set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 metric 1

set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 metric 10
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 2 metric 1
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface ge-0/0/4.0 level 2 metric 100
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0


regress@R2> show configuration protocols rsvp |display set
set protocols rsvp interface all

regress@R2> show configuration protocols mpls |display set
set protocols mpls label-switched-path red_lsp to 4.4.4.4
set protocols mpls label-switched-path red_lsp ldp-tunneling
set protocols mpls label-switched-path red_lsp primary via_R3
set protocols mpls label-switched-path blue_lsp to 4.4.4.4
set protocols mpls label-switched-path blue_lsp ldp-tunneling
set protocols mpls label-switched-path blue_lsp primary via_R5
set protocols mpls path via_R3 21.2.3.2 strict
set protocols mpls path via_R3 21.3.4.2 strict
set protocols mpls path via_R5 21.2.5.2 strict
set protocols mpls path via_R5 21.4.5.1 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols isis interface lo0.0


regress@R2> show configuration protocols ldp |display set
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

set routing-options router-id 2.2.2.2
```

R3 Configuration:

```
regress@R3> show configuration protocols isis |display set
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/0.0 level 2 metric 1
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 metric 1
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 2 metric 10
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface ge-0/0/4.0 level 2 metric 100
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0


regress@R3> show configuration protocols ldp |display set
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

set protocols mpls interface all
set protocols mpls interface fxp0.0 disable

set protocols rsvp interface all

set routing-options router-id 3.3.3.3
```

R4 Configuration:

```
regress@R4> show configuration protocols isis|display set
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 metric 1
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 2 metric 100
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface ge-0/0/4.0 level 2 metric 10
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface ge-0/0/6.0 level 2 metric 1
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0


regress@R4> show configuration protocols ldp |display set
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

set protocols mpls interface all
set protocols mpls interface fxp0.0 disable

set protocols rsvp interface all
set routing-options router-id 4.4.4.4
```

R5 Configuration:

```
regress@R5> show configuration protocols isis|display set

set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 metric 100
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 2 metric 1
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface ge-0/0/4.0 level 2 metric 10
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface ge-0/0/6.0 level 2 metric 1
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0


regress@R5> show configuration protocols ldp |display set
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

set protocols mpls interface all
set protocols mpls interface fxp0.0 disable

set protocols rsvp interface all

set routing-options router-id 5.5.5.5
```

R6 Configuration:

```
regress@R6> show configuration protocols isis|display set
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface ge-0/0/6.0 level 2 metric 1
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0


regress@R6> show configuration protocols ldp |display set
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

set protocols mpls interface all
set protocols mpls interface fxp0.0 disable

set routing-options router-id 6.6.6.6

set routing-options autonomous-system 60030

set protocols bgp group R0 type internal
set protocols bgp group R0 local-address 6.6.6.6
set protocols bgp group R0 family inet unicast
set protocols bgp group R0 neighbor 100.100.100.100
set protocols bgp group R0 export bgp_export_policy

set protocols bgp group R1 type internal
set protocols bgp group R1 local-address 6.6.6.6
set protocols bgp group R1 family inet unicast
set protocols bgp group R1 neighbor 1.1.1.1
set protocols bgp group R1 export bgp_export_policy

set protocols bgp group R1 neighbor 1.1.1.1
set protocols bgp group R7 type internal
set protocols bgp group R7 local-address 6.6.6.6
set protocols bgp group R7 family inet unicast
set protocols bgp group R7 neighbor 7.7.7.7


set policy-options policy-statement bgp_export_policy term one from route-
filter 121.1.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term one then next-hop 6.6.6.6
set policy-options policy-statement bgp_export_policy term one then accept

set policy-options policy-statement bgp_export_policy term two from route-
filter 121.2.0.0/16  orlonger
set policy-options policy-statement bgp_export_policy term two then next-hop 6.6.6.6
set policy-options policy-statement bgp_export_policy term two then accept
```

R7 Configuration:

```
shregress@R7> show configuration protocols isis |display set
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface ge-0/0/5.0 level 2 metric 1
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface ge-0/0/6.0 level 2 metric 1
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0


regress@R7> show configuration protocols ldp |display set
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

set protocols mpls interface all
set protocols mpls interface fxp0.0 disable

set protocols bgp group R0 type internal
set protocols bgp group R0 local-address 7.7.7.7
set protocols bgp group R0 family inet unicast
set protocols bgp group R0 export bgp_export_policy
set protocols bgp group R0 neighbor 100.100.100.100
set protocols bgp group R1 type internal
set protocols bgp group R1 local-address 7.7.7.7
set protocols bgp group R1 family inet unicast
set protocols bgp group R1 export bgp_export_policy
set protocols bgp group R1 neighbor 1.1.1.1
set protocols bgp group R6 type internal
set protocols bgp group R6 local-address 7.7.7.7
set protocols bgp group R6 family inet unicast
set protocols bgp group R6 neighbor 6.6.6.6


set routing-options router-id 7.7.7.7
set routing-options autonomous-system 60030

set policy-options policy-statement bgp_export_policy term one from route-
filter 131.1.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term one then next-hop 7.7.7.7
set policy-options policy-statement bgp_export_policy term one then accept
set policy-options policy-statement bgp_export_policy term two from route-
filter 131.2.0.0/16 orlonger
set policy-options policy-statement bgp_export_policy term two then next-hop 7.7.7.7
set policy-options policy-statement bgp_export_policy term two then accept
```

Enabling ISIS-SR and Stitching to RSVP

R0 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa
set protocols isis source-packet-routing sensor-based-stats per-interface-per-member-link ingress
set protocols isis source-packet-routing sensor-based-stats per-interface-per-member-link egress
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 100

set protocols isis interface ge-0/0/1.0 level 2 post-convergence-lfa
set protocols isis interface ge-0/0/2.0 level 2 post-convergence-lfa
```

R1 Configuration:

```
set protocols isis backup-spf-options use-post-convergence-lfa
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 101
```

R2 Configuration:

```
regress@R2> show configuration protocols isis|display set
set protocols isis backup-spf-options use-post-convergence-lfa

set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 102
```

R3 Configuration:

```
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 103
```

R4 Configuration:

```
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 104
```

R5 Configuration:

```
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 105
```

R6 Configuration:

```
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 106
```

R7 Configuration:

```
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 107
```

LDPoSRTE Configurations

Base configuration is the same as the previous chapter, enabling ISIS SR on R2, R3, R4, and R5.

R2 Configuration:

```
regress@R2> show configuration protocols isis|display set
set protocols isis backup-spf-options use-post-convergence-lfa

set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 102
```

R3 Configuration:

```
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 103
```

R4 Configuration:

```
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 104
```

R5 Configuration:

```
set protocols isis source-packet-routing sensor-based-stats per-sid ingress
set protocols isis source-packet-routing sensor-based-stats per-sid egress
set protocols isis source-packet-routing srgb start-label 200000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv4-index 105
```

Appendix B: LDP to SR Migration Using SR-TE Policy

By Colby Barth and Jordan Stewart, Juniper Networks

This appendix explores how to incrementally migrate from LDP to SR using SR-TE policies. The end result is not just migration to "segment routing" but a migration to using SR-TE policies that set a foundation for taking advantage of many of SR's innovations.

The topology shown in Figure B.1 is used here. The network is using ISIS for internal reachability advertisements and LDP as the label distribution protocol.

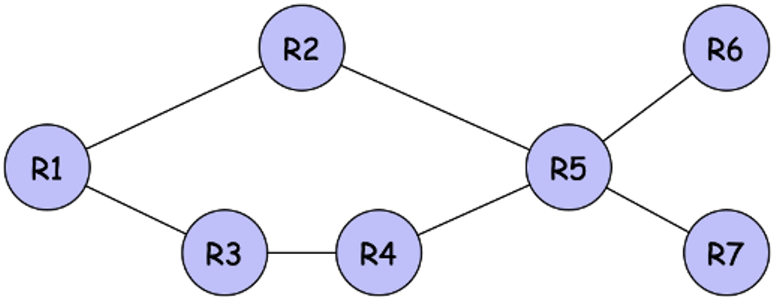


Figure B.1 Example ISIS/LDP Network

Before beginning, let's verify the route on R1 for the R6 destination:

```
regress@R1> show route table inet.3 1.1.1.6

inet.3: 6 destinations, 12 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.6/32         *[LDP/9] 00:01:59, metric 30
                   >  to 10.1.12.2 via ge-0/0/1.0, Push 26
```

As you can see in the output, R1 can forward a packet to R6 by encapsulating it with the 26. The route is present in the inet.3 table as a LDP route with a preference of 9.

Enabling Segment-Routing Extensions

Now let's enable segment-routing extensions in ISIS so the routers can exchange SIDs. Since the example network is quite small, we can reserve a small label range (100) and the index for each router will simply be the router's number. For example, R1's index equals 1, R2's index equals 2, etc. Our resulting network topology will look like Figure B.2.

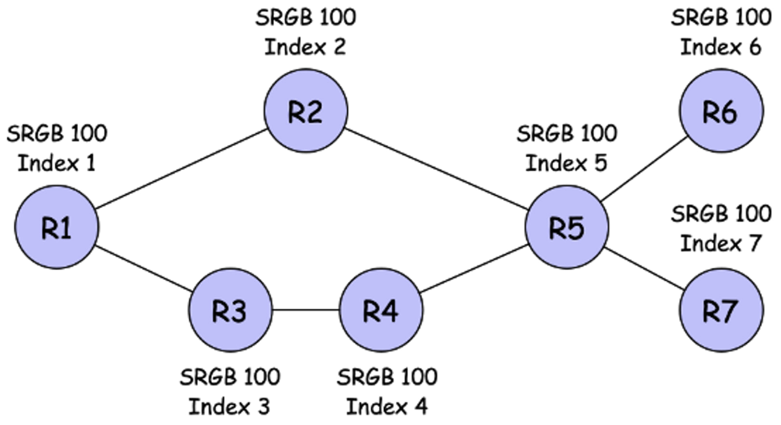


Figure B.2: Example Network with SR Enabled

The added segment-routing specific configuration on R1 is:

```
[edit protocols isis]
source-packet-routing {
    node-segment ipv4-index 1;.
    srgb start-label 100 index-range 100;
```

Once segment-routing extensions are enabled, exchanging SIDs can be verified with various operational CLI. For example, `show isis adjacency detail` or `show isis database extensive`, but more importantly for our example, let's have another look at the route to R6 on R1:

```
regress@R1> show route table inet.3 1.1.1.6

inet.3: 6 destinations, 12 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.6/32         *[LDP/9] 00:01:59, metric 30
                   >  to 10.1.12.2 via ge-0/0/1.0, Push 26
                   [L-ISIS/14] 00:01:59, metric 30
                   >  to 10.1.12.2 via ge-0/0/1.0, Push 106
```

You can see that a new L-ISIS route has been added to the inet.3 RIB but by default, LDP is still preferred due to its better route preference than the newly added SR route. Notice that the SR route pushes label 106 for packets to reach R6, this will be an interesting detail later.

The Role of SR-TE Policies in Migration

As you just saw, enabling SR IGP extensions has little impact on the basic forwarding behavior of our sample network. The network is still using the LDP FECs/labels for forwarding due to the better route preference. Now let's say you want to migrate all traffic from R1 to R6 to start using the SR label of 106. Said another way, you want to migrate all traffic with the 1.1.1.6 protocol next-hop to SR. Let's add a SR-TE-policy.

Before we add SR-TE-policies to the picture, Junos maintains multiple Traffic Engineering Databases (TEDs). The l3-unicast-topology TED is used by SR-TE to translate SIDs to IP addresses for static segment-lists as well as for calculating a Directed Acyclic Graph (DAG) to facilitate dynamic path computation. To populate the l3-unicast-topology, the following CLI knob is required:

```
[edit protocols isis]
traffic-engineering {
    l3-unicast-topology;
```

Now, back to adding that SR-TE-policy. Let's add a policy on R1 to R6. We'll leverage a dynamic computation service by adding the `compute` keyword to the primary segment-list, (see Figure B.3). Dynamic compute can refer to local compute-profiles in the event more sophisticated constraint-based paths are desired.

NOTE A complete description of this capability can be found in the Juniper TechLibrary: https://www.juniper.net/documentation/en_US/junos/topics/topic-map/segment-routing-lsp-configuration.html#id-enabling-distributed-cspf-for-segment-routing-lsps.

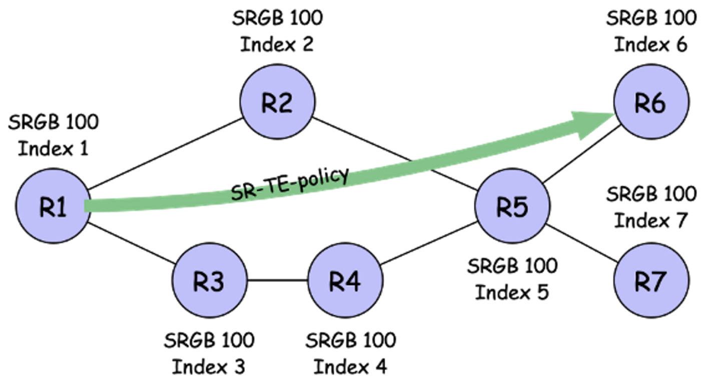


Figure B.3 SR-TE-policy from R1 to R6

Our example SR-TE-policy on R1 is shown here:

```
[edit protocols source-routing]
source-routing-path r1-to-r6 {
    to 1.1.1.6;
    primary {
        sl1 {
            compute;
```

Now let's look at the same route on R1:

```
regress@R1# run show route table inet.3 1.1.1.6

inet.3: 6 destinations, 13 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
1.1.1.6/32        *[SPRING-TE/8] 00:01:49, metric 1, metric2 30
                  >  to 10.1.12.2 via ge-0/0/1.0, Push 106
                  [LDP/9] 00:01:49, metric 30
                  >  to 10.1.12.2 via ge-0/0/1.0, Push 26
                  [L-ISIS/14] 00:01:49, metric 30
                  >  to 10.1.12.2 via ge-0/0/1.0, Push 106
```

Notice that the `SPRING-TE` route is now preferred, but to the better protocol preference, and it's using the same prefix SID/label as the L-ISIS route. Thus it's taking the shortest path just like the L-ISIS route, and just like the LDP tunnel did. Now we have incrementally migrated to using `segment-routing` for all packets from R1 to R6. This process can be repeated on a destination-by-destination basis as needed. But what if you want to be more granular?

Getting More Granular with Colors

In the previous example, we migrated all the traffic from R1 to R6 to SR, but what if we wanted to migrate some specific services while leaving others to remain on LDP for a time? You can use the color attribute to facilitate such a desire. Let's add a color (123) to our previous SR-TE-policy, shown in Figure B.4 (all other 'things' are left the same as in Figure B.3 for easy comparison).

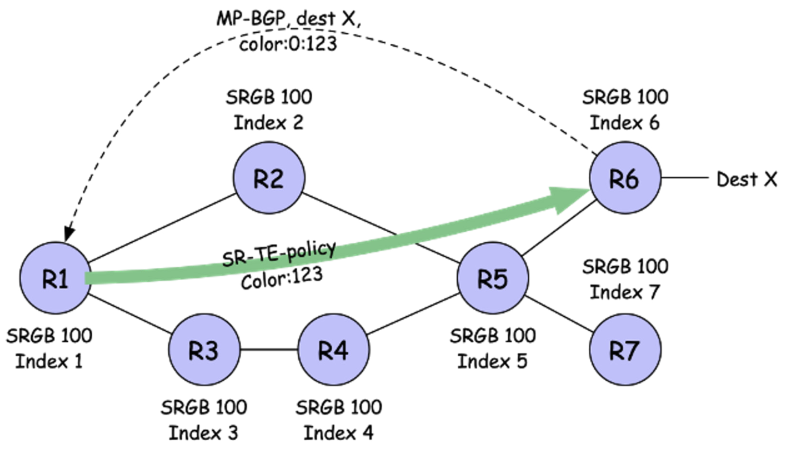


Figure B.4 Colored Service Prefixes

And here’s the configuration:

```
[edit protocols source-routing]
source-routing-path r1-to-r6 {
    to 1.1.1.6;
    color 123;
    primary {
        sl1 {
            compute;
```

Now let’s have another look at that same route, 1.1.1.6, on R1:

```
regress@R1# run show route table inet.3 1.1.1.6

inet.3: 6 destinations, 12 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.6/32         *[LDP/9] 00:05:38, metric 1
                   >  to 10.1.12.2 via ge-0/0/1.0, Push 26
                   [L-ISIS/14] 00:05:38, metric 30
                   >  to 10.1.12.2 via ge-0/0/1.0, Push 106
```

Notice the route is now gone from inet.3 and has been put into the inetcolor.0 RIB. It still uses the same node SID/label as the original L-ISIS route:

```
regress@R1# run show route table inetcolor.0

inetcolor.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.6-123<c>/64
                   *[SPRING-TE/8] 00:01:48, metric 1, metric2 30
                   >  to 10.1.12.2 via ge-0/0/1.0, Push 106
```

Now you can add a BGP extended color community to a prefix/service-route that results in that route resolving over inetcolor.0 routes with the same color instead of inet.3. As mentioned before, this would facilitate a migration on a service-route by service-route basis.

MORE? Advertising service-routes with BGP extended community colors documentation can be found here: https://www.juniper.net/documentation/en_US/junos/topics/concept/color-based-mapping-of-vpn-services-overview.html.

That covers two options for incremental transition from LDP to segment-routing using SR-TE policies. But what about automation? Could the manual creation of many SR-TE policies present a somewhat cumbersome barrier?

SR-TE Policy Automation

Junos has the ability to dynamically create SR TE policies when a BGP service route resolves over a particular Protocol Next Hop (a PE router). This reduces the amount of manual configuration needed on the ingress routers. The online documentation for dynamic tunnels can be found here: https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/dynamic-tunnels-edit-routing-options.html.

This example builds on the previous examples by creating `dynamic-tunnel` templates for the colored SR TE policies. Before we jump into the details it's worth noting that the use of colors is optional for dynamic-tunnels, just as it is optional for manually configured SR TE policies. Thus, if granular service route mapping/migration is undesirable, the color attribute and associated configuration can be omitted.

In Figure B.5 there are service routes being announced from both R6 and R7 to R1 with the color community 123 attached to them.

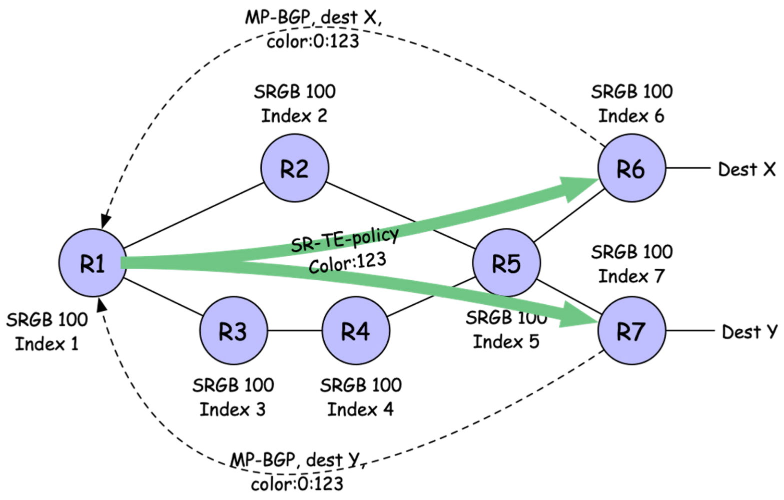


Figure B.5 Service-mapping Over SR-TE Policies with Dynamic Tunnels

Let's add the following `dynamic-tunnels` template to the configuration to trigger the dynamic creation of the SR-TE policies:

```
[edit routing-options]
dynamic-tunnels {
    dyn-sr-te-lsps {
        spring-te {
            source-routing-path-template {
                sr-te-template color 123;
            }
            destination-networks {
                1.1.1.0/24;

[edit protocols source-routing]
source-routing-path-template sr-te-temp {
       primary {
         sl1 {
              compute;
```

The configuration results in the dynamic creation of shortest path SR-TE policies when service routes arrive from the service route protocol next hops. You can see that the dynamic-tunnels have been created as they are maintained in a dynamic-tunnel database:

```
regress@R1> show dynamic-tunnels database terse
*- Signal Tunnels #- PFE-down
Table: inetcolor.0

Destination-network: 1.1.1.0-0<c>/24
Destination         Source      Next-hop              Type       Status
1.1.1.6-123<c>/64   -           1.1.1.6:64:sr-te-temp spring-te Established
1.1.1.7-123<c>/64   -           1.1.1.7:64:sr-te-temp spring-te Established
```

And further, now you can also bring up the SPRING-TE routes in inetcolor.0 as done in the previous examples. Since this example included service routes from R7, you also see a route to 1.1.1.7 with the label 107 corresponding to the SID advertised via segment-routing IGP extensions:

```
regress@R1> show route table inetcolor.0

inetcolor.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.0-0<c>/24
                 *[Tunnel/305] 2d 21:12:26
                   Tunnel
1.1.1.6-123<c>/64
                 *[SPRING-TE/8] 00:01:48, metric 1, metric2 30
                  >  to 10.1.12.2 via ge-0/0/1.0, Push 106
1.1.1.7-123<c>/64
                 *[SPRING-TE/8] 00:01:49, metric 1, metric2 30
                 >  to 10.1.12.2 via ge-0/0/1.0, Push 107
```

Summary

This short appendix explored a few scenarios for incrementally migrating from LDP to SR using SR-TE policies. It covered the relative priorities between route entries of different protocols, service-mapping with BGP extended community colors, and, finally, how to automate the creation of SR-TE policies. The end result is not just a migration to SR but also setting a foundation to take advantage of many of SR's innovations in the TE space.

Appendix C: SR Auto-bandwidth Using Express-Segments

By Colby Barth and Jordan Stewart, Juniper Networks

Let's start by looking at a pretty simple MPLS auto-bandwidth deployment depicted in Figure C.1. The network topology is the classic *Fish Network* and the operator has chosen to deploy MPLS auto-bandwidth so that the resources of the network can be used efficiently based on the incoming traffic towards network destinations.

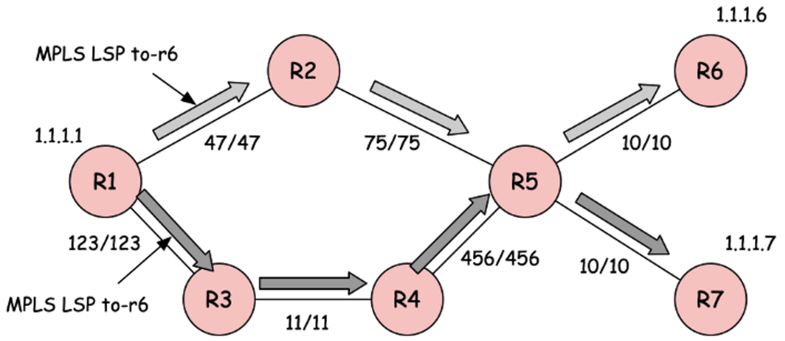


Figure C.1 *MPLS Auto-bandwidth LSPs on FishNet*

There are two LSPs configured on R1; one to R6 and another to R7. The LSPs are configured as auto-bandwidth LSPs. The #/# values on each link in Figure C.6 represent the IGP/TE metrics of the links.

Example MPLS Auto-bandwidth Configuration

Let's show the base configuration:

```
regress@R1> show configuration protocols mpls
statistics {
    auto-bandwidth;
}
label-switched-path to-r7 {
    to 1.1.1.7;
    auto-bandwidth;
    }
}
label-switched-path to-r6 {
    to 1.1.1.6;
    auto-bandwidth;
```

The MPLS LSPs use the local TED, shown in Figure C.2, to calculate routes for the LSPs. Notice that *all* the nodes and links of the topology shown in Figure C.1 above are represented in the local TED. The IGP and TE metrics are deployed symmetrically and equally, as depicted in Figure C.1 above, so as to represent the distance between routers for facilitating least delay routing. The IGP and TE metrics can be seen in the detailed view of each of the TE-link in the local TED, shown in Figure C.2.

Note that there are two TED instances in Junos:

- Default Traffic Engineering TED Instance: This is populated by ISIS, OSPF, and BGP-LS (via TED-export policy), and is used by the compute-engine that computes RSVP-TE paths.
- L3-Unicast Topology instance: This is populated by ISIS, OSPF, BGP-LS (via TEDexport policy), BGP-EPE and express-segments, and the DB is used by the compute-engine that computes SR-TE paths.

Here's the local traffic-engineering database:

```
regress@R1> show ted link topology-type traffic-engineering
ID                      ->ID                        LocalPath LocalBW
R1.00(1.1.1.1)            R3.00(1.1.1.3)                  0 0bps
R1.00(1.1.1.1)            R2.00(1.1.1.2)                  0 0bps
R2.00(1.1.1.2)            R1.00(1.1.1.1)                  0 0bps
R2.00(1.1.1.2)            R5.00(1.1.1.5)                  0 0bps
R3.00(1.1.1.3)            R1.00(1.1.1.1)                  0 0bps
R3.00(1.1.1.3)            R4.00(1.1.1.4)                  0 0bps
<truncated>
```

And here's a detailed look at the TE-link connect R1 to R3:

```
regress@R1> **show ted link topology-type traffic-engineering detail**
R1.00(1.1.1.1)->R3.00(1.1.1.3), Local: 10.1.13.1, Remote: 10.1.13.2
  Local interface index: 335, Remote interface index: 334
  LocalPath: 1, Metric: 10, IGP metric: 10, StaticBW: 1000Mbps, AvailBW: 1000Mbps
           Color: 0 <none>
  localBW [0] 0bps     [1] 0bps     [2] 0bps     [3] 0bps
  localBW [4] 0bps     [5] 0bps     [6] 0bps     [7] 0bps
  AvailBW [0] 500Mbps  [1] 500Mbps  [2] 500Mbps  [3] 500Mbps
  AvailBW [4] 500Mbps  [5] 500Mbps  [6] 500Mbps  [7] 500Mbps
<truncated>
```

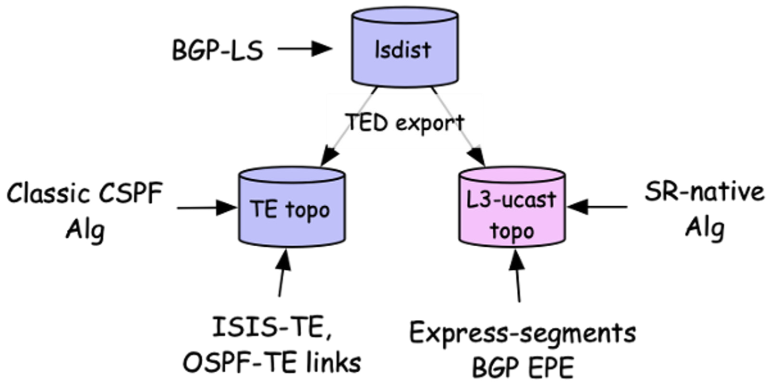


Figure C.2 TE Topology in Junos

Creating a Customized Topology with Express-segments

Now let's use `express-segments` to automatically create an abstraction of the network using policy over which segment-routing policies can be deployed. It involves creating a simple abstraction using the two underlay auto-bandwidth LSPs. The abstraction will interconnect R1 to R6 and R7, inheriting the cumulative IGP, TE metrics, and the current bandwidth of the underlay paths/LSPs. Furthermore, the policy will define two sets of express segments; one set will be colored using admin-group red (10) and the other colored with admin-group blue (20). The resulting customized topological representation of the network is shown Figure C.3.

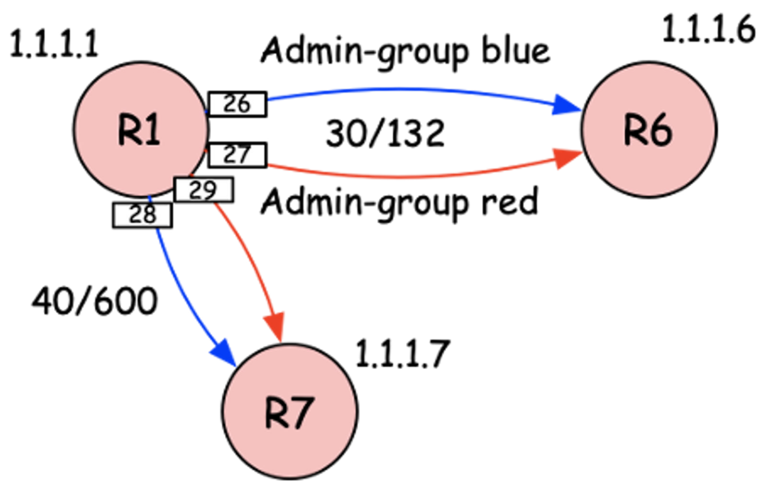


Figure C.3 Abstract TE Topology Created by the Express-segments

Figure C.3's topology is created using three configurations on R1:

- Creating admin-groups used to color the express-segments
- Policy for creating the express-segments
- Express-segment templates for calling the policy and creating the te-links

Here's the admin-group configuration example:

```
regress@R1> show configuration protocols mpls
admin-groups {
    red 10;
    blue 20;
```

Here's the policy for matching underlay paths:

```
regress@R1> show configuration policy-options
policy-statement all-lsps {
    then accept;
```

And the automatic express-segment creation example:

```
regress@R1> show configuration protocols express-segments
segment-template blue {
    admin-group blue;
    }
}
segment-template red {
    admin-group red;
    }
}
segment-set auto-create-red {
    membership-policy all-lsps;
    template {
        red;
    }
}
segment-set auto-create-blue {
    membership-policy all-lsps;
    template {
        blue;
    }
}
traffic-engineering;
```

You can see in the next output of `show express-segments` the adj-SID/label assigned to the te-link along with the symbolic name, link IDs, etc. You can also see the newly created l3-unicast topology database that contains these express-segment te-links and a detailed view of one reveals the inherited attributes from the underlay paths and the manually assigned colors:

```
regress@R1> show express-segments
To                Segment     Link        Status        Elapsed    Segment
                  Label       LocalID                   Time       Name
--                --------    --------    ------        -------    -------
1.1.1.6           26          2147483659  Up (T)       16:44:32    auto-create-blue-1.1.1.6
1.1.1.7           28          2147483661  Up (T)       16:44:32    auto-create-blue-1.1.1.7
1.1.1.6           27          2147483660  Up (T)       16:44:32    auto-create-red-1.1.1.6
1.1.1.7           29          2147483662  Up (T)       16:44:32    auto-create-red-1.1.1.7
```

Resulting L3 Unicast Topology Database

Remember the two TED instances in Junos? You can now see that the l3-unicast topology database is now populated with express-segments:

```
regress@R1> show ted link topology-type l3-unicast
ID                         ->ID                          LocalPath LocalBW
1.1.1.1                      1.1.1.6                             0 0bps
1.1.1.1                      1.1.1.6                             0 0bps
1.1.1.1                      1.1.1.7                             0 0bps
1.1.1.1                      1.1.1.7                             0 0bps
```

Detailed view of the ‘blue’ express-link from R1 to R6

```
regress@R1> show ted link topology-type l3-unicast detail
1.1.1.1->1.1.1.6, Local: 1.1.1.1, Remote: 1.1.1.6
  Local interface index: 2147483659, Remote interface index: 0
  Link name: auto-create-blue-1.1.1.6
  LocalPath: 0, Metric: 132, IGP metric: 30, StaticBW: 500Mbps, AvailBW: 500Mbps
      Color: 0x100000 blue
  localBW [0] 0bps  [1] 0bps  [2] 0bps  [3] 0bps
  localBW [4] 0bps  [5] 0bps  [6] 0bps  [7] 0bps
  AvailBW [0] 500Mbps  [1] 500Mbps  [2] 500Mbps  [3] 500Mbps
  AvailBW [4] 500Mbps  [5] 500Mbps  [6] 500Mbps  [7] 500Mbps
  IPV4 P2P-Adj-SID SID: 26 Flags: 0x30 Weight: 1
<truncated>
```

Ingress SR-TE Policy Creation

Once you've created a simple abstraction (l3-unicast topology) of the network, you can use it for calculating segment-lists for SR policies. Let's explore a couple of interesting SR policies that can be created. First, let's create a policy that will use any red te-link to get to R6. In this example we create a compute-profile to associate with a SR policy that constrains the segment-list calculation to only red links in the l3-unicast topology. We also assign a color attribute of 10 to the SR policy to facilitate service-mapping.

NOTE In the compute-profile you need to disable label compression, since in this example R1 does not have a prefix SID for R6 because no SR IGP extensions have been enabled in the network.

Example SR policy to select red links to R6

```
regress@R1> show configuration protocols source-packet-routing
compute-profile default-red {
    admin-group include-any red;
    no-label-stack-compression;
    }
}
source-routing-path to-r6-red {
    to 1.1.1.6;
    color 10;
    primary {
        sl1 {
            compute {
                default-red;
```

You can now look at the resulting SR policy in more detail to see what has been calculated. Notice that a single segment-list has been computed and the top-most label is 27. Label: 27 is the label that was assigned to the express-segment shown before. The calculated metrics are also those inherited by the express-segment:

```
regress@R1> show spring-traffic-engineering lsp name to-r6-red detail
Name: to-r6-red
  Tunnel-source: Static configuration
  To: 1.1.1.6-10<c>
  State: Up
    Path: sl1
    Outgoing interface: NA
    Auto-translate status: Disabled Auto-translate result: N/A
    Compute Status:Enabled , Compute Result:success , Compute-Profile Name:default-red
    Total number of computed paths: 1
    Computed-path-index: 1
      BFD status: N/A BFD name: N/A
      TE metric: 132, IGP metric: 30; Metric optimized by type: TE
      computed segments count: 1
```

```
          computed segment : 1 (computed-adjacency-segment):
          label: 27
          source router-id: 1.1.1.1, destination router-id: 1.1.1.6
          source interface-address: 1.1.1.1, destination interface-address: 1.1.1.6
```

You can also look at the resulting `SPRING-TE` route, added to the inetcolor.0 RIB, to verify that service route resolution is as expected. In the following output you can see that something looks different! *Why is the label operation not 'push 27'?*

```
regress@R1> show route table inetcolor.0

inetcolor.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.6-10<c>/64
                   *[SPRING-TE/8] 03:22:55, metric 1, metric2 30
                    >  to 10.1.12.2 via ge-0/0/1.0, Push 21
```

The `Push 21` operation for the `SPRING-TE` route is using the express-segment that is an abstraction of the underlay MPLS LSP from R1 to R6 (Figure C.4). If you look at the MPLS LSP and the mpls.0 RIB you can see where the label 21 comes from.

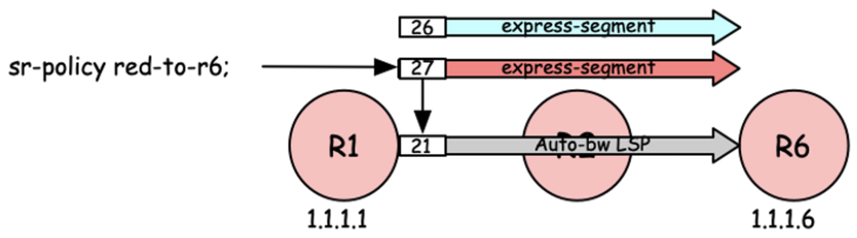


Figure C.4 *Sr-policy Via express-segment Over MPLS LSP*

```
regress@R1> show route table mpls.0 protocol express-segments

mpls.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

27                 *[EXPRESS-SEG/6] 03:29:03, metric 1
                    >  to 10.1.12.2 via ge-0/0/1.0, Swap 21

regress@R1> show mpls lsp detail
Ingress LSP: 2 sessions

1.1.1.6
  From: 1.1.1.1, State: Up, ActiveRoute: 0, LSPname: to-r6, LSPid: 7
 <snip>
    Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 30)
 10.1.12.2 S 10.1.25.2 S 10.1.56.2 S
    Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt):
          10.1.12.2(Label=21) 10.1.25.2(Label=23) 10.1.56.2(Label=3)
```

Summary

Junos express-segments, https://datatracker.ietf.org/doc/draft-saad-sr-fa-link/, offer a powerful migration option through underlay topology abstraction. Abstraction allows the use of a policy to represent the potential ability to interconnect many endpoints. Thus, abstraction does not necessarily offer all possible connectivity options, but presents a view of potential connectivity according to the policies that determine how the domain's administrator wants to allow the domain resources to be used.