JUNIPER NETWORKS | Engineering Simplicity

Segment Routing User Guide

Published
2025-12-16

JUNOS

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Segment Routing User Guide

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

About This Guide

Segment routing (SR) is a method of generating a series of instructions that indicate how a packet can be forwarded or processed across a topology. These instructions are called segments. Use this guide to learn about segment routing and configure segment routing (SR-MPLS or SRv6) on your network.

We also support Segment Routing Traffic Engineering (SR-MPLS and SRv6) and micro-SIDs. See the Feature Explorer tool for all the platforms that support SR-TE and micro-SIDs.

This guide does not cover SR-TE in detail and micro-SIDs at all. Stay tuned for documentation updates on SR-TE and micro-SIDs!

1

CHAPTER

Segment Routing Overview

IN THIS CHAPTER

Introduction to Segment Routing

IN THIS SECTION

What Is Segment Routing and Source Packet Routing in Networking?

SUMMARY

In a segment-routed network, devices generate a series of instructions that represent the ways that a packet can be forwarded or processed. These instructions are called segments. By stacking segments together, operators can create precise paths between any two devices. This enables features such as a BGP-free core, traffic engineering (TE), backup paths, multi-domain paths, and multi-topology networking. In the data plane, segments can be represented using MPLS labels or IPv6 addresses.

IN THIS SECTION

Prerequisite Knowledge

We assume that the readers understand general IP routing and MPLS label switching principles and have a strong working knowledge of either IS-IS or OSPF. We've covered these concepts in other resources in our Technical Documentation.

Segment routing (SR) is a method of generating a series of instructions that indicate how a packet can be forwarded or processed across a topology. These instructions are called segments. These instructions

can then be written onto a packet so that transit devices forward and process the packet as intended. Multiple segments can be stacked together to define an end-to-end path between any two devices.

Segments are advertised directly inside routing protocols such as IS-IS, OSPF, and BGP. This means there is no need to run an additional protocol to advertise these segments throughout the network. This section defines the concept of a segment. This section also covers how segments are advertised, and how they can be used to create a precise path across a network. We further explore how these segments can be represented in the data plane by using MPLS labels or by using special IPv6 addresses called SRv6 addresses. These options are called SR-MPLS and SRv6, respectively.

Examples of Instructions Represented by Segments

A segment is the name given to a construct that represents one single instruction. This could be an instruction related to forwarding, or it could be an instruction relating to how the packet should be processed locally at the receiving router.In segment routing, these instructions are advertised throughout the network—for example, throughout an IS-IS level or an OSPF area. The result is that any SR-capable device can stack these instructions together to create a precise path across the network to any arbitrary remote device.

Here are a few examples of instructions that a segment can represent:

- Forward a packet down the interior gateway protocol (IGP) shortest path to a particular remote router.
- Instruct a transit router to send the packet directly to a next-hop neighbor, and therefore out of a specific local interface.
- Instruct an autonomous system (AS) border router to send the packet toward a specific external BGP (EBGP) peer, overriding the BGP best-path decision.
- Load-balance a packet between two or more transit or endpoint routers.
- Send a packet towards the nearest border router out of multiple potential exit points.
- Instruct a transit router to redirect the packet down a specific traffic-engineered (TE) tunnel.
- Define a smaller constrained topology within your main topology, and then forward a packet down the shortest path within that constrained topology.
- Remove all the segment instructions that have been pushed onto the packet, and then process the packet itself in a specific Layer 2 or Layer 3 forwarding table.

Segments Are Advertised Inside Your Existing Routing Protocols

In an SR network, each device generates a variety of segments that represent how it can forward or process a packet. The majority of these segments are then advertised to all other routers inside the same routing domain, such as the same OSPF area or IS-IS level, or between BGP peers.

Instead of using a dedicated protocol, segment routing advertises segments directly in IS-IS, OSPF, and BGP through protocol extensions. You don't need to enable new protocols for segment routing. You can simply rely on the protocols that you already use.

As a result, every SR-enabled device in your routing domain has full visibility of every segment instruction that has been advertised by every other SR-enabled node in the network. In other words, every device learns about every advertised segment that every other device can receive.

Figure 1 on page 4 demonstrates this idea and shows a high-level example of what segments look like in an SR-MPLS network. The concept is similar in SRv6.

Figure 1: Each router generates an OSPF LSA or IS-IS TLV containing their topology information. This object also advertises segment instruction information.

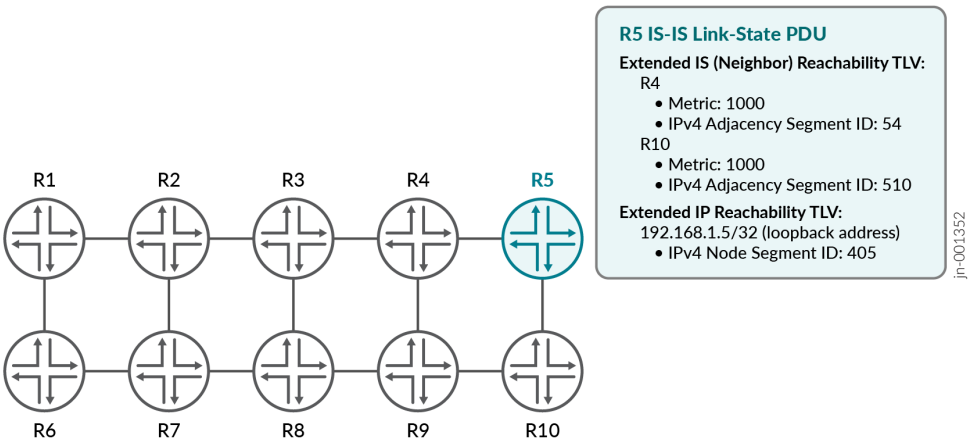


Figure 1 on page 4 shows a topology of 10 routers, numbered R1 to R10. In a link-state network, every router generates an object that describes its place in the network. OSPF calls this object a link-state advertisement (LSA), while IS-IS calls it a link-state protocol data unit (LSP). In both cases, SR can append additional information to this LSA or LSP.

For example, in Figure 1 on page 4, R5 has announced that it has adjacencies to R4 and R10. Segment routing can tag this information in the LSA or LSP with a segment instruction that says R5 can use segment routing to send traffic directly to these neighbors, along with a number called the segment

identifier (SID). These are called adjacency segments in SR-MPLS. The equivalent construct in SRv6 is called an End.X SID.

R5 has also tagged its own loopback IPv4 address prefix with an instruction that indicates R5 is willing to accept traffic that has been sent down the IGP shortest path. This is called a node segment in SR-MPLS.

The LSA or LSP is flooded throughout the routing domain, which means that every router in the OSPF area or IS-IS level learns about every single segment that has been advertised using these protocols. Using this segment information, SR capable routers can stack one or more segment instructions directly onto a packet to indicate the path that the packet should take to a particular remote router, along with information about the way that the packet should be processed.

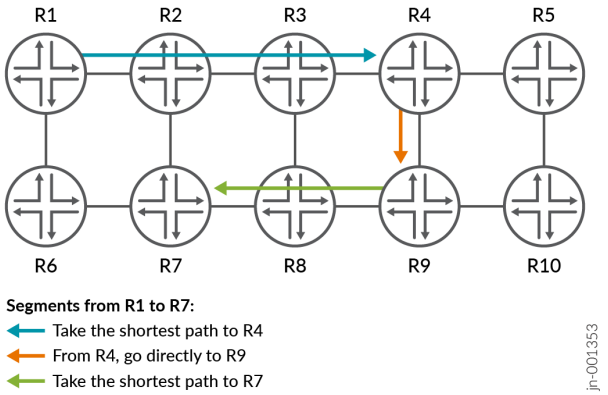
Combine Segments to Create a Precise Path Across a Network

After every router has announced the various segments that represent the instructions it has generated, any router in the network can combine these segments together to create any arbitrary path of your choice. This may involve creating a stack of segments, or it may be as simple as using one single shortest-path segment to the destination router.

Segment routing achieves this without the need to actively signal or request the path, because all of the segment instructions have already been advertised and learned by existing routing protocols. Any router can simply calculate the required path, extract the required segments associated with that path, and then append those instructions onto the packet in the data plane.

Figure 2 on page 5 shows a human-readable example of how these segments can be combined together.

Figure 2: Segment routing builds instructions to send traffic over the topology.

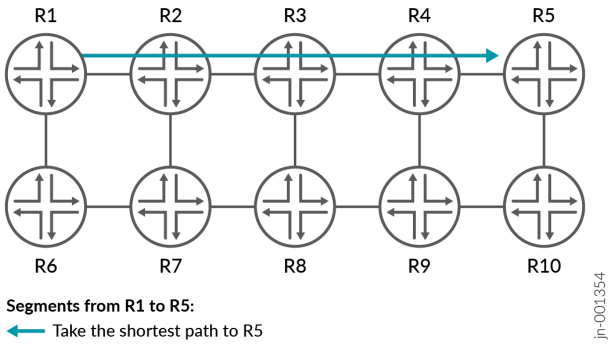


Starting from R1, the figure shows a series of three instructions that send a packet towards R7. The first segment follows the shortest path to R4. The second segment goes directly from R4 to R9. The final instruction takes the shortest path to R7.

R4 generated and advertised the first two segments. R7 generated the third segment. Any router in the network can then take advantage of these segments to create a precise path across a topology.

You don't need to use a stack of segments. For example, if your only requirement is for traffic to follow the shortest path to a remote device, then the packet will require only one segment. This is shown in Figure 3 on page 6, where only one segment is required to describe the shortest path between R1 and R5.

Figure 3: Only one segment is required to describe the shortest path between any two devices.



This is enough to enable a classic BGP-free core design without traffic engineering, where packets simply follow the IGP shortest path to the destination border router.

Using MPLS or IPv6 to Represent Segments in the Data Plane

In the data plane, segments can be represented using MPLS labels or IPv6 addresses. These methods are called SR-MPLS and SRv6, respectively.

Figure 4 on page 7 shows a high-level overview of what each protocol looks like in the data plane. SR-MPLS uses one MPLS label to represent a single segment, while SRv6 can encode one or more segments inside a single IPv6 address. When multiple IPv6 addresses are required to describe the full path, these addresses are stored inside an IPv6 extension header called the Segment Routing Header (SRH).

Figure 4: A high-level comparison of creating a stack of SR-MPLS and classic SRv6 segments to create a traffic-engineered path.

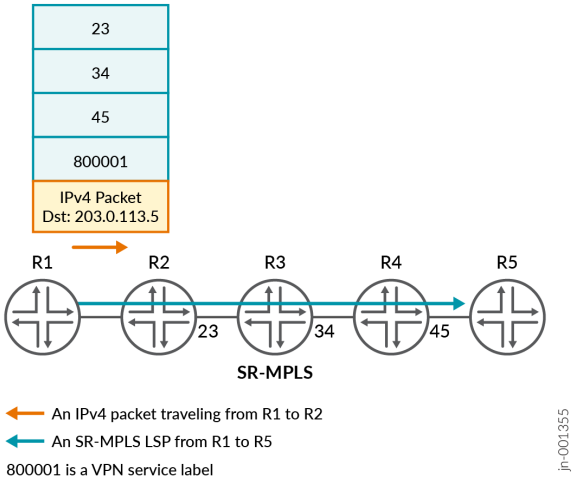


Figure 4 on page 7 demonstrates a stack of segments in each data plane protocol. However, if a packet simply needs to follow the shortest path to a remote device, then it is not necessary to use a stack of segments. Instead, this instruction can be encoded inside a single MPLS label or IPv6 address.

SR-MPLS and SRv6 both use a numerical construct called a segment identifier (SID). SID is used to communicate the MPLS label or IPv6 address that should be used in the data plane. The SID value is used in different ways depending on the type of segment being advertised.

For example, in SR-MPLS, the SID value is sometimes the exact MPLS label that a router expects to receive for that segment. However, other segment types use the SID value as an index that can be combined with a pre-allocated set of MPLS labels to calculate the correct MPLS label that a router expects to receive for a segment.

In contrast, SRv6 deployments tend to allocate an entire unique subnet of IPv6 addresses to each individual router in the network. Routers then allocate individual segments from this larger prefix. The segments and the prefix are both advertised around the network in such a way that any router can, at the very least, follow the metrically shortest path toward that segment.

Both IS-IS and OSPF create and advertise the same type of segments, and they both use the segment ID value in the same way. However, each protocol has its own unique way of advertising this information.

Refer to our documentation that explores SR-MPLS and SRv6 separately and in greater detail, along with the IS-IS and OSPF configuration that enables these data plane protocols and the unique segment types offered by SR-MPLS and SRv6. Before reading this detailed documentation, readers are strongly advised to first read the documents mentioned in the section below to gain a broad understanding of segment routing in general.

What's Next

We strongly recommend reading the following articles for a general understanding of segment routing:

- "What Features and Designs Does Segment Routing Enable?" on page 8
- "What Is Source Routing in a Segment Routed Network?" on page 15
- "How Do Routers Decide What Segments to Push onto a Packet in Segment Routing?" on page 17

What Features and Designs Does Segment Routing Enable?

SUMMARY

Segment routing enables many useful features, including a BGP-free core, traffic engineering (TE), local repair backup paths, anycast routing, and multi-topology networking. All of these features require no additional signaling, and are powered by existing routing protocols. Additionally, centralized controllers can create bandwidth reservations and traffic-engineered paths across IGP areas and levels, and BGP autonomous systems.

IN THIS SECTION

Prerequisite Knowledge

We assume that you've read "What Is Segment Routing and Source Packet Routing in Networking?" on page 2, along with the prerequisite topics.

Segment routing (SR) offers many powerful features that add tremendous value to a variety of networks, including service provider networks, data center networks, mobile networks, and large enterprise networks.

Some of these features are exclusive to segment routing. Existing path protocols and tunneling protocols such as RSVP and LDP also offer some other features. Even in this case, segment routing offers new advantages and enhancements when compared to existing options.

This document provides a high-level overview of these features and a broad understanding of the benefits of deploying segment routing. We also offer various documents that explain each of these features in greater technical detail, including their configuration and verification.

All the features described in this guide are available for SR-MPLS. Most features are also available for SRv6.

Automatic Full-Mesh of Shortest-Path Tunnels to Other SR-Enabled Devices

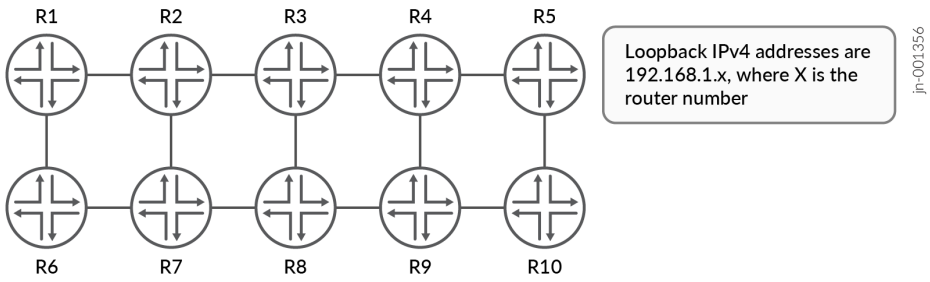
When an operator runs MPLS and LDP on Junos OS routers, the result is that each device automatically creates a full mesh of label-switched paths (LSPs) to every other LDP device in the network. These LSPs simply follow the IGP shortest path to the destination. This is enough to power a BGP-free core, or VPN services such as Ethernet VPN (EVPN) and Layer 3 VPNs (L3VPN).

Segment routing automatically creates shortest-path tunnels to all other SR-enabled devices, for both SR-MPLS and SRv6.

In an SR-MPLS network, the inet.3 routing tables can be automatically populated with IPv4 LSPs to all other SR-MPLS enabled devices. This is true even in the most basic of SR-MPLS deployments. If you also run IPv6 in your network, it is also easy to populate the inet6.3 table with IPv6 LSPs.

Figure 5 on page 9 demonstrates this concept. It shows a network of ten routers, numbered R1 to R10.

Figure 5: SR-MPLS Network: Automatic Full Mesh of Shortest-Path LSPs from R1 to Other Routers. This is verified in the CLI output below:



After enabling SR-MPLS with IS-IS, the output below shows that router R1's inet.3 table contains a full mesh of LSPs to the other nine routers. SR-MPLS automatically creates this full mesh, requiring no additional signaling.

```
user@R1> show route table inet.3

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

192.168.1.2/32     *[L-ISIS/14] 00:06:45, metric 100
                    >  to 10.1.2.2 via ge-0/0/0.0
192.168.1.3/32     *[L-ISIS/14] 00:02:18, metric 200
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300403
192.168.1.4/32     *[L-ISIS/14] 00:02:18, metric 300
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300404
192.168.1.5/32     *[L-ISIS/14] 00:02:18, metric 400
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300405
192.168.1.6/32     *[L-ISIS/14] 00:06:45, metric 100
                    >  to 10.1.6.6 via ge-0/0/2.0
192.168.1.7/32     *[L-ISIS/14] 00:02:18, metric 200
                       to 10.1.2.2 via ge-0/0/0.0, Push 300407
                    >  to 10.1.6.6 via ge-0/0/2.0, Push 300407
192.168.1.8/32     *[L-ISIS/14] 00:02:18, metric 300
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300408
                       to 10.1.6.6 via ge-0/0/2.0, Push 300408
192.168.1.9/32     *[L-ISIS/14] 00:02:18, metric 400
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300409
                       to 10.1.6.6 via ge-0/0/2.0, Push 300409
192.168.1.10/32    *[L-ISIS/14] 00:02:18, metric 500
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300410
                       to 10.1.6.6 via ge-0/0/2.0, Push 300410
```

Note that the outgoing MPLS label appears to be predictable for each remote node. This is not the default behavior of SR-MPLS, but it is very easy to enable with just a few lines of configuration. Note that the LSPs to R7, R8, R9, and R10 automatically take advantage of equal-cost multipath (ECMP) options.

In an SRv6 network, if all the devices run SRv6, then you can create an automatic full mesh of tunnels to all other SRv6-enabled devices. However, there is no strict requirement to run SRv6 on all of your transit devices. For example, if an older transit device cannot run SRv6, then these devices can still be enabled for regular IPv6. This will be enough for these routers to offer transit functionality to the SRv6 tunnels in

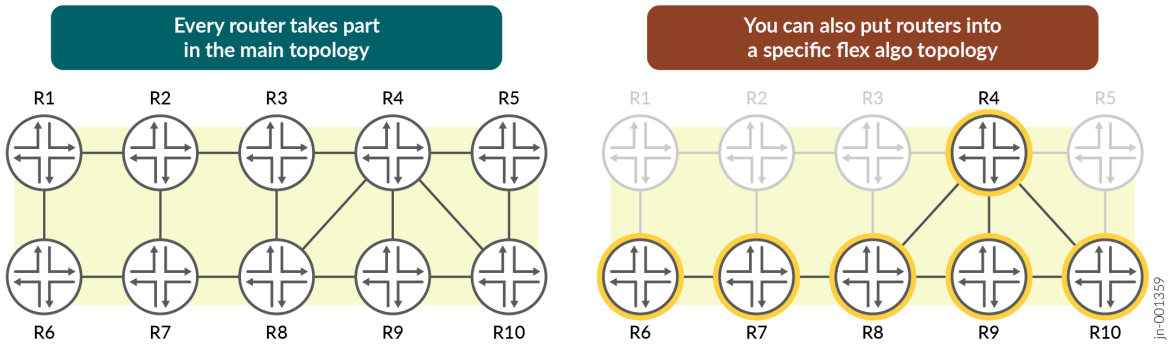
your network. In this case, you'll have at least a partial mesh of tunnels to every other SRv6-enabled device.

Flexible Algorithm

Flexible Algorithm or Flex Algo is a scalable method of creating multitopology networks. Using admin groups, you can tag certain links as belonging to a particular Flex Algo topology. New segment instructions are then created that are unique to that topology. This guarantees that traffic will only ever be contained within your topology. The traffic will never leak out to routers that are not part of the topology because the devices inside the topology routes the traffic within the topology.

Figure 6 on page 11 demonstrates this concept. Every single router takes part in the main topology. Then, certain links can additionally take part in one or more smaller topologies.

Figure 6: Every router takes part in the main topology. You can then choose to place routers and links into a specific flex algo topology.



Traditional multi-topology methods required great effort to configure and maintain. In contrast, using admin groups gives you the option to reuse admin groups between topologies, or to choose whether to exclude or include one or more admin groups.

Each topology is associated with a numerical color. This enables Flex Algo to take advantage of the BGP color community to automatically bind a prefix to a particular topology. This offers a much more scalable solution than historical methods of binding a prefix to a specific multi-topology network.

Topology-Independent Loop-Free Alternate (TI-LFA)

TI-LFA creates local repair backup paths that can be used immediately in times of link or node failure. There are other protocols that also offer local repair paths. However, some of these protocols can only offer limited topology coverage due to the requirement to avoid backup paths that might create a loop.

Other protocols can cover the entire network, but each backup path needs to be set up and maintained separately.

In contrast, TI-LFA offers 100% topology coverage with no additional signaling. By pushing a series of segments onto a packet, topological loops are completely avoided. In addition, TI-LFA backup paths are identical to the path that will be used once the network has finished converging from the link or node failure. This is called the post-convergence path. This means traffic doesn't need to shift a second time to a new post-convergence path once it's moved to a backup path. This reduces network jitter.

When we pre-install these backup paths into the forwarding plane, we often observe that we can reduce downtime to as little as 50ms during the switchover.

You can also use the TI-LFA backup paths to protect plain IP traffic. Further, Flex Algo topologies automatically ensure that TI-LFA backup paths are contained only within the topology that you have designed. TI-LFA is a powerful feature, and is one of the primary drivers that network operators consider when choosing whether to deploy segment routing.

Microloop Avoidance

In link-state networks, different routers learn and process topology change updates at different times. This can create temporary loops during network convergence events. For example, if one router converges before a neighboring router, and decides that its neighbor is the next best hop, then the neighboring router might send the packet back again if the neighbor has not yet finished converging. These loops are known as microloops.

Microloops are brief, millisecond events. However, in the modern era, a substantial quantity of traffic can transit across a link in that time. Microloops have the power to temporarily flood a link with traffic, which could potentially cause a complete outage on that link during this time.

Microloop avoidance (MLA) is feature in segment routing that identifies areas in your topology where microloops might occur. Then, when a link or node fails, Junos OS can temporarily push segment instructions to ensure that remote routers will forward the traffic correctly, even if they have not yet finished converging to the new topology. These segments are often used for no more than a few seconds, but this can be enough to prevent a catastrophic outage during the convergence process.

Anycast

If you configure a prefix on two or more routers, then both devices can announce a segment that represents the instruction to send traffic down the shortest path toward that prefix. This may be a prefix on a shared point-to-point or broadcast link, or it could be a /32 IPv4 or /128 IPv6 address that is configured on two or more devices.

As a result, you can send traffic down the shortest path toward the metrically closest router. Or, if all devices have equal cost, you can load-balance traffic between them..

This feature is known as anycast routing. It is common in the core of a network. By configuring two or more core routers to announce the same segment, you can create a TE path that also has the ability to take advantage of equal-cost paths when they are available. This is not possible in protocols such as RSVP, where you need to create two or more separate explicit paths if you want to load-balance across those paths.

A less common but powerful example is to configure two or more router borders to announce the same prefix and segment. If you use this shared segment as a BGP protocol next-hop, you can then create a design where traffic is simply transported to the closest egress point.

On-Demand Next-Hops

Traffic engineering historically required network operators to configure individual TE paths on each ingress PE router toward every egress router. Many of these TE paths shared similar TE characteristics. For example, every single TE path might be configured to use TE metrics instead of IGP metrics, or to avoid an admin group that indicates whether a link will soon undergo maintenance.

As an alternative to individually defining all the endpoints to your paths on each PE router, segment routing offers the ability to automatically detect when a TE path should be created. This feature is called On-Demand Next-Hops (ODN).

When you run ODN in an SR network, paths are automatically calculated and built to any valid BGP protocol next-hop on any valid BGP prefix. The path to the remote PE is then calculated using D-CSPF, based on constraints of your choice. This offers a solution that is much more scalable than manually defining all of your endpoints individually.

This feature is also available in RSVP. Once again though, in a segment routed network, you don't need to signal or maintain these paths.

Controller-Based Traffic Engineering

Using an external controller (such as Juniper Paragon Pathfinder), you can run a protocol called Path Computation Element Protocol (PCEP) that can communicate directly with every SR enabled device in your network. The controller can calculate TE paths on behalf of all the devices in the network, and then use PCEP to write those paths directly to the ingress router of that path.

One or more SR-enabled routers can advertise topology information to the controller using BGP Link-State (BGP-LS). This enables the controller to learn your topology without also becoming a part of the same link-state topology. The controller can then automatically generate a graphical view of your topology and the tunnels that travel across that topology.

If you have multiple routing domains (such as different OSPF areas, IS-IS levels, or autonomous systems), then you can configure devices in each of these domains to advertise the topology to your controller. As a result, the controller can calculate paths across domain boundaries, and then inform an ingress router

of the exact segments that should be written onto the packet, regardless of how many routing domains the tunnel travels across.

The controller can also offer additional services that are not offered by segment routing itself, such as bandwidth reservations on a per-path basis. The controller can also fully optimize paths in response to network events, such as moving a less important tunnel on one device to another path to make room for a higher priority tunnel on another device.

Color-Aware Traffic Engineering

Network operators often create multiple tunnels between two endpoints so that each tunnel can follow a different path based on the importance of the traffic. For example, delay-sensitive traffic might take a shorter path than best-effort traffic. Similarly, best-effort traffic might be deliberately routed away from the physically shortest links in your network, so that delay-sensitive traffic can take the quickest path without competing with the best-effort traffic.

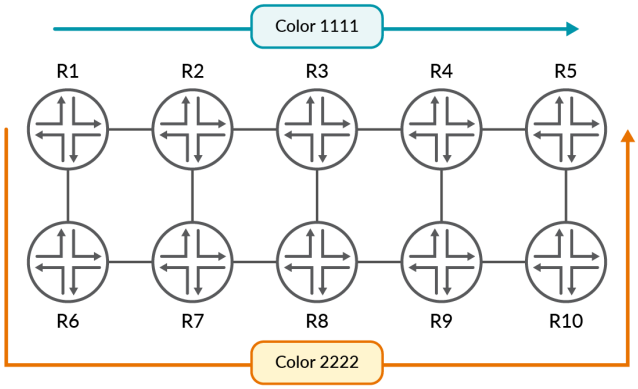
Historically, this feature requires the use of complex routing policies to map individual BGP prefixes to particular paths. These routing policies often need to be manually maintained, and can easily be misconfigured.

In contrast, segment routing offers you the ability to tag your tunnels with a color, which is simply a numerical identifier that represents the intent of the tunnel. For example, you might choose to assign color value 10 to any tunnel on any router that carries best-effort traffic. As another example, you might use color value 20 to represent tunnels that carry delay-sensitive traffic.

You can then take advantage of the BGP color community to automatically bind a learned prefix to a TE tunnel with a matching color. This removes the traditional complexity of manually creating routing policies to achieve the same goal. It also offers you a much more scalable method of deploying multiple paths to a destination.

Figure 7 on page 15 demonstrates this concept.

Figure 7: Two paths From R1 to R5 With Each Path Tagged With a Different Numerical Color



R1 has two paths to the remote router R5. One path is used by delay-sensitive traffic, and the other is used for traffic destined to the public Internet. When R5 advertises BGP prefixes that are delay sensitive, R5 can attach BGP color community value 1111 to these prefixes. R1 will then automatically associate these prefixes with the path that has a matching color value.

Further reading

- "What Is Source Routing in a Segment Routed Network?" on page 15
- "How Do Routers Decide What Segments to Push onto a Packet in Segment Routing?" on page 17

What Is Source Routing in a Segment Routed Network?

SUMMARY

In a segment-routed network, source routing enables the first router in a path to calculate the end-to-end path. The source router can then write a series of MPLS labels or IPv6 addresses to the packet that represent the segment instructions for the path. Transit routers do not need to perform any routing calculations on an incoming packet. Instead, they simply follow the instructions made by the source router.

Prerequisites Knowledge

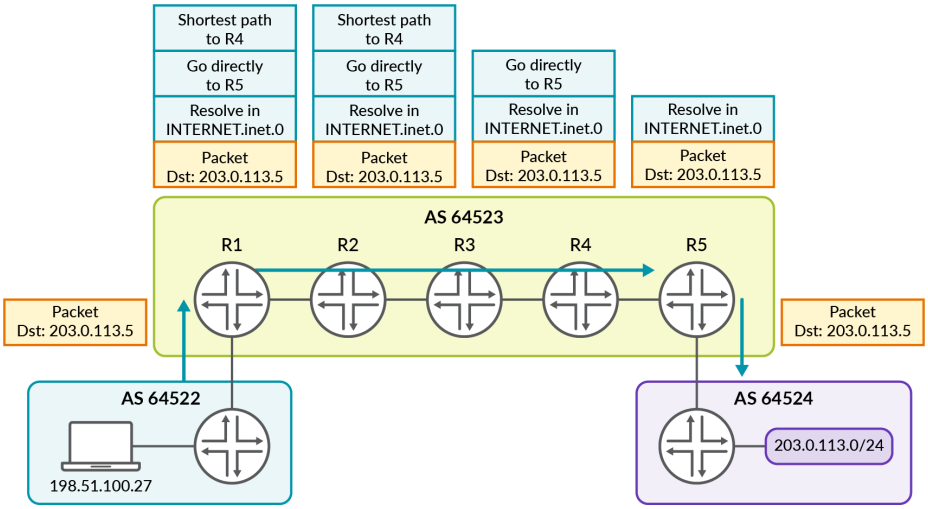
We assume that readers have read the sections titled "What Is Segment Routing and Source Packet Routing in Networking?" on page 2 and "What Features and Designs Does Segment Routing Enable?" on page 8, along with the prerequisites for these topics.

Segment routing is also known as Source Packet Routing in Networking (SPRING).

Source routing does not mean forwarding packets based on the source IP address. This concept is called source-based routing. Source routing refers to the ability of the first node in any arbitrary path to calculate the path across the topology. The first node in the path is the source router. This source router performs the routing decisions for every single hop. The transit routers in the path then simply follow the instructions that they receive.

For example, in Figure 8 on page 16, you can see an autonomous system (AS) that is part of a larger end-to-end path between the ultimate source and destination of a packet. AS 64523 is just one of three networks that a packet must traverse to arrive at its destination.

Figure 8: From the Perspective of AS 64523, R1 is the Source Router of the Segment-Routed Path.



Notes:

Each arrow represents a path within an autonomous system, or between autonomous systems. In AS 64523, this path is a traffic-engineered segment-routed path from R1 to R5.

Each path is tagged with a unique numerical color. R1 uses this color value to bind certain BGP prefixes to a specific path.

jn-001346

From the perspective of segment routing inside AS 64523, the source router is the ingress border router R1, and the destination router is the egress border router R5 at the other side of the AS.

In a network with segment routing, R1 will have a series of pre-calculated paths to various other routers in the network, such as R5. This may be as simple as an MPLS LSP or SRv6 tunnel that follows the

metrically shortest path. Or, it can be a traffic-engineered path that follows a series of strict and loose hops. Either way, R1 is responsible for calculating the routing decisions for the entire path. After R1 calculates the path, it writes the required instructions onto the packet. This allows the packet to traverse your network precisely as you want.

In this situation, source of the path makes routing decisions. Transit routers then simply forward traffic based on the decisions that have already been made by the source router. This differs from a regular IP network where each transit router makes its own local routing decision at every hop. As such, in a segment-routed network, the source router performs source routing.

How Do Routers Decide What Segments to Push onto a Packet in Segment Routing?

SUMMARY

In a segment-routed network, nodes can push one or more segments onto a packet to create a precise forwarding path. If a packet simply follows the shortest path, then Junos OS decides the correct segment automatically. If the path involves traffic engineering (TE), then you can configure the segments manually or calculate them automatically according to your TE constraints. You can also use a centralized controller to calculate this information.

IN THIS SECTION

Prequisite Knowledge

We assume that the readers have read the topics titled "What Is Segment Routing and Source Packet Routing in Networking?" on page 2, "What Features and Designs Does Segment Routing Enable?" on page 8, and "What Is Source Routing in a Segment Routed Network?" on page 15, along with the prerequisites for these topics.

In a segment-routed network, Junos OS determines the SR-MPLS or SRv6 segments required to create forwarding instructions through a combination of automatic calculation and manual configuration. Paths can utilize a single MPLS label or SRv6 address, or a stack of segments.

This document briefly introduces you to all of the options. These options are then explained in greater detail in separate documents. This document uses SR-MPLS examples, but you can apply the same concepts to SRv6.

SPF Shortest Paths and Flex Algo Shortest Paths

If your only requirement is for traffic to follow the IS-IS or OSPF metrically shortest path to a remote router, then Junos OS simply uses the segment associated with that instruction. This happens automatically and is true for both regular topologies and Flexible Algorithm (Flex Algo) topologies.

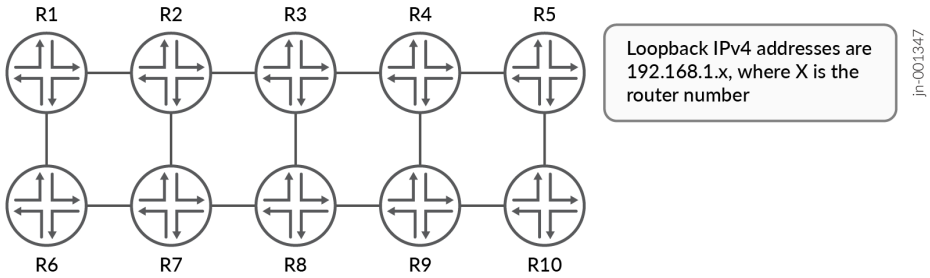
In the case of the default complete topology, Junos OS will have a full mesh of SR tunnels in its inet.3 table (for SR-MPLS) or in its inet6.3 table (for SRv6).

Recall that the inet.3 and inet6.3 tables store all of the operational label-switched paths (LSP) that ingress on this device. By default, this table is used by BGP as a potential way to resolve BGP protocol next-hops. Every tunnel in these tables contains information about the shortest-path segment that must be pushed onto the packet to transport the traffic to the destination of the tunnel.

If you choose to enable any Flex Algo topologies in your network, then Junos OS also creates a separate full mesh of LSPs for all devices inside that topology. Depending on how you enable this feature, each topology can have its own equivalent of an inet.3 table on each Junos OS router, with a full mesh of shortest-path tunnels to every other router in the topology.

To understand this concept, see Figure 9 on page 18 and CLI output 1 below. First, Figure 9 on page 18 shows a network of ten routers, numbered from R1 to R10. Each router has a loopback IPv4 address of the format 192.168.1.x, where X is the router number.

Figure 9: A Topology of 10 Routers, Numbered R1 to R10



CLI output 1 below focuses on router R1's inet.3 routing table after you enable a basic SR-MPLS deploymenton all devices. It shows a full mesh of LSPs to every other device in the network. If you configure shortest path segments on each node in your network, then this full mesh of tunnels is created automatically on every SR-MPLS device.

Output 1: After enabling a basic SR-MPLS deployment on all routers, R1's inet.3 table shows a full mesh of label-switched paths to all other routers

```
user@R1> show route table inet.3

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

192.168.1.2/32     *[L-ISIS/14] 00:06:45, metric 100
                    >  to 10.1.2.2 via ge-0/0/0.0
192.168.1.3/32     *[L-ISIS/14] 00:02:18, metric 200
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300403
192.168.1.4/32     *[L-ISIS/14] 00:02:18, metric 300
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300404
192.168.1.5/32     *[L-ISIS/14] 00:02:18, metric 400
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300405
192.168.1.6/32     *[L-ISIS/14] 00:06:45, metric 100
                    >  to 10.1.6.6 via ge-0/0/2.0
192.168.1.7/32     *[L-ISIS/14] 00:02:18, metric 200
                       to 10.1.2.2 via ge-0/0/0.0, Push 300407
                    >  to 10.1.6.6 via ge-0/0/2.0, Push 300407
192.168.1.8/32     *[L-ISIS/14] 00:02:18, metric 300
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300408
                       to 10.1.6.6 via ge-0/0/2.0, Push 300408
192.168.1.9/32     *[L-ISIS/14] 00:02:18, metric 400
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300409
                       to 10.1.6.6 via ge-0/0/2.0, Push 300409
192.168.1.10/32    *[L-ISIS/14] 00:02:18, metric 500
                    >  to 10.1.2.2 via ge-0/0/0.0, Push 300410
                       to 10.1.6.6 via ge-0/0/2.0, Push 300410
```

Each of these LSPs contains a single outgoing transport label. The label represents the instruction to follow the shortest path to that remote router. SR-MPLS calculates the automatically and requires no additional configuration on R1.

For example, the output contains an ingress LSP to R5 with a destination address of 192.168.1.5. If R1 learns a BGP prefix with a protocol next-hop of 192.168.1.5, R1 resolves the prefix to this LSP. Therefore, R1 automatically uses the shortest-path segment associated with that tunnel. In this case, R1

pushs a single SR-MPLS transport label of 300405. This is the label that R2 receives when the you send traffic down the shortest path towards R5.

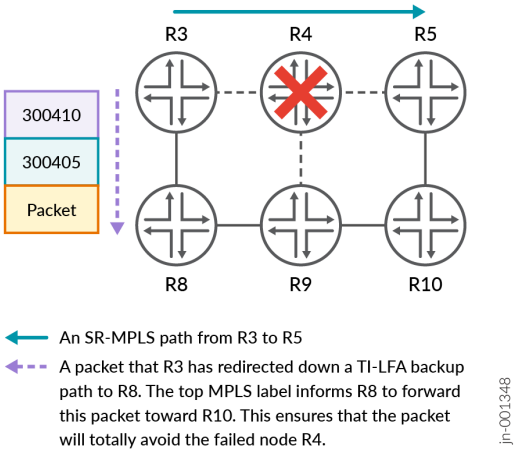
Topology-Independent Loop-Free Alternate Backup Paths and Microloop Avoidance

A similar automatic result takes place when you enable Topology-Independent Loop-Free Alternate (TI-LFA) or when you run microloop avoidance. In addition to segments that represent the shortest path to a remote device, segment routing also offers segments that forward traffic directly from one specific router to a next-hop neighbor. By combining shortest-path segments and direct next-hop segments, you can create any arbitrary path in the network. This approach prevents temporary loops that can occur during network convergence events. TI-LFA uses these two segment types to calculate and build backup paths.

When SPF calculates the best path to a destination link or router, TI-LFA reruns SPF to determine a backup path that protects against local link or neighboring router failures. Once TI-LFA calculates the path, it automatically decides the segment stack required on this path. This stack can range from a single shortest-path segment to a more complex series of hop-by-hop instructions. Either way, this happens without any additional user intervention.

Figure 10 on page 20 shows six routers where R4 has failed.

Figure 10: Routers Automatically Calculate the Correct Segments Required for TI-LFA Backup Paths and Microloop Avoidance.



In a TI-LFA network, R3 is prepared for this event by calculating a backup path that routes around R4. However, R3 has also identified that R8 and R9 are both potential sources for a loop. To avoid this problem, R3 automatically pushes a second segment onto the packet that instructs the network to send the traffic to R10. From R10, the packet can then safely be delivered to R5. These segments are

calculated and pushed automatically and require no additional user intervention beyond enabling TI-LFA. The same is true of microloop avoidance. In the event of a topology change, SPF calculates a new best path. If microloop avoidance detects the potential for a brief microloop on the new path, then Junos OS automatically pushs the necessary segments onto the packet that ensures that the packet reaches its destination.

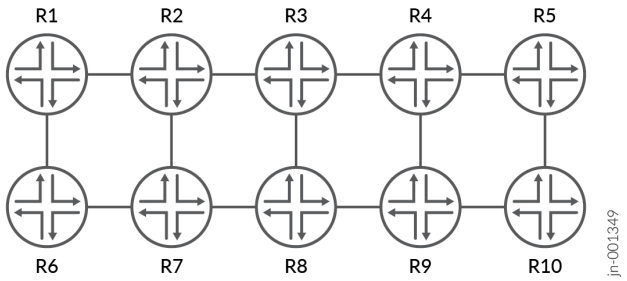
Traffic Engineering with Manual User Segment Configuration

Junos OS offers several methods for determining the segments written to a packet for a traffic-engineered path. The simplest, but least scalable, method is for a user to manually create a tunnel to a remote router and then configure the MPLS labels or SRv6 addresses required for that path. In this case, the router does not actually check whether the segments are valid and correct. A few segments are not advertised at all, and cannot be verified for accuracy. In this case, Junos OS simply follows your instructions and writes whatever you specify to the packet.

Figure 11 on page 21 shows an example of this. Using the same topology of 10 routers, it introduces you to a Junos OS configuration construct called a segment list. By defining a named segment list, you can define the precise segments that should be written to a packet. You can then refer to the segment list when you create an SR-TE label-switched path.

The example uses SR-MPLS, but the concept is identical for SRv6.

Figure 11: A Segment List That Defines a Strict Path From R1 to R10. The Final Two Digits Each of Each Label Represent the Source and Destination Router of That Path.




```
protocols {
    source-packet-routing {
        segment-list PATH_R1_R5_EXPLICIT_LABELS {
            HOP_1 label 1004012;
```

```
            HOP_2 label 1004023;
            HOP_3 label 1004034;
            HOP_4 label 1004045;
        }
    }
}
```

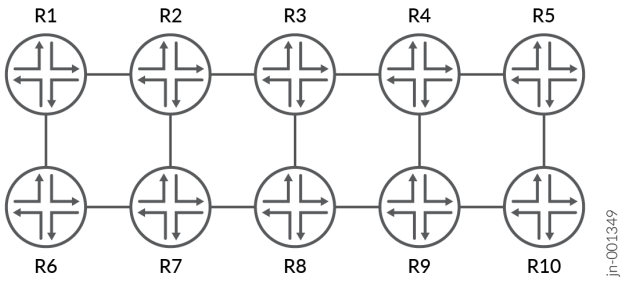
This method is easy to configure but, like all static paths, cannot react to topology changes. If one of the links or nodes represented by your segments fail, then the entire path fails. Another disadvantage of this option is that it cannot inherently identify failures along the path. However, you can enable a new version of Bidirectional Forwarding Detection (BFD) called Seamless BFD (S-BFD) that actively monitors the path. This shuts the path down when a failure is detected enabling you to move over to another backup or secondary path.

Traffic Engineering With Manual User Configuration and IP Lookups

A more scalable method of traffic engineering is to configure the required IP hops that the traffic engineered path must take. For example, a path can combine loopback IP addresses and network interface addresses of the devices in the path. Junos OS can then convert these addresses into shortest-path instructions and direct next-hop instructions, and then calculate the required MPLS label or SRv6 address for that segment.

Figure 12 on page 22 demonstrates this concept.

Figure 12: A Segment List That Defines a Path From R1 to R10. Junos OS Converts These IP Addresses into Equivalent SR-MPLS or SRv6 Segments.




```
protocols {
    source-packet-routing {
```

```
        segment-list PATH_R1_R10_INTERFACE_IP {
            auto-translate;
            HOP_1 ip-address 10.1.6.6;
            HOP_2 ip-address 10.6.7.7;
            HOP_3 ip-address 10.7.8.8;
            HOP_4 ip-address 10.8.9.9;
            HOP_5 ip-address 10.9.10.10;
        }
    }
}
```

This is similar to the example in Figure 11 on page 21 except that this time, the named segment list refers directly to interface IP addresses and loopback IP addresses. When this segment list is used in an SR-TE LSP, Junos OS automatically converts the IP addresses to the required segments. Note that this method still requires you to define the path manually. As such, this path cannot respond to the failure of any of the devices in this path. Nevertheless, the automatic translation of IPs to segments does introduce a small amount of dynamic behavior when compared to defining the segment values explicitly in your configuration.

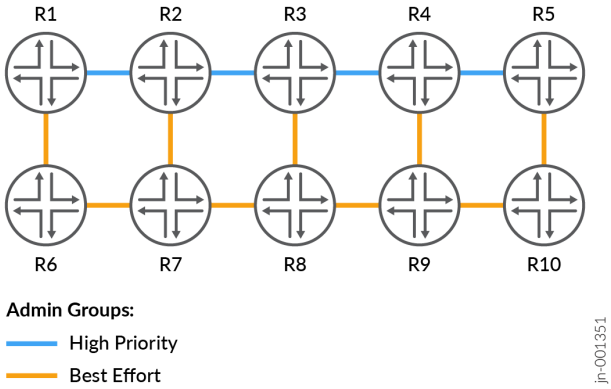
Traffic engineering With Dynamic Path Calculation

An even more scalable method is to calculate the path dynamically based on a set of traffic engineering constraints. Junos OS will then calculate the correct segments for the calculated path. This option uses a version of the Constrained Shortest Path First (CSPF) algorithm used by RSVP. The algorithm is called Distributed CSPF (D-CSPF). The name reflects the fact that the protocol can be run on any router in the network. This is in contrast to the option of using a centralized controller.

D-CSPF offers some TE constraints found in RSVP, such as admin groups and shared risk link groups. Additionally, it includes segment routing-specific constraints, like limiting the maximum number of segments pushed onto a packet or exclusively using links that support local repair. D-CSPF can also consider alternative shortest-path metrics, such as TE metrics or delay metrics. Junos OS collects a set of constraints together into an object called a compute profile. This set of constraints can then be applied consistently to any TE path that requires those constraints.

Figure 13 on page 24 briefly introduces this concept.

Figure 13: A Junos OS Compute Profile That Contains Your Traffic Engineering Constraints. Junos OS Uses This to Calculate the Correct SR-MPLS or SRv6 Segments for Your TE Tunnels.




```
protocols {
    source-packet-routing {
        compute-profile ONLY_USE_HIGH_PRIORITY {
            admin-group include HIGH_PRIORITY;
            maximum-segment-list-depth 6;
            metric-type {
                te;
            }
        }
    }
}
```

The Figure 13 on page 24 shows that each link in the network has been tagged with one of two admin groups, called HIGH_PRIORITY and BEST_EFFORT. The diagram also shows the configuration for a basic compute profile, which calculates a path that only considers links that are tagged with the admin group HIGH_PRIORITY. This example also shows that you can include other traffic engineering constraints inside one compute profile object. After defining a compute profile, you can use this object in two ways. The first option is to reference it inside a tunnel with a destination endpoint that you have manually defined. In this case, although the endpoint is configured manually, everything else is calculated dynamically by Junos OS, including the path and the segment list. If the topology changes, Junos OS automatically calculates a new path and a new segment list. The second option is to reference the compute profile as part of an On Demand Next-Hops (ODN) deployment. This feature does not require you to manually define each tunnel endpoint. Instead, Junos OS can dynamically build new tunnels to any valid BGP protocol next-hop.

In both cases, the result is that Junos OS decides the segments that should be written to a packet.

Traffic engineering with a Centralized Controller

You can use a central controller to calculate and create a segment list. A controller also offers you all of the TE options mentioned earlier: creating manual paths with explicit segments or IP hops, and dynamic paths based on TE constraints. In addition, controllers offer a variety of powerful features, such as topology visibility, graphical path visibility, bandwidth constraints, and TE between routing domains. Once the controller determines the path that a tunnel should take, the controller finds the segments associated with that path, and then writes this information directly to the data plane of the ingress router for the path. The path can easily cross a domain boundary. The ingress router does not need to be aware of every single segment in the path. The ingress router need to know only the MPLS labels or IPv6 addresses that should be written to the packet, along with the direct next-hop for the traffic.

Further reading

The above sections are a few basic concepts of Segment Routing Traffic Engineering. Stay tuned for detailed documentation updates for Segment Routing Traffic Engineering in upcoming revisions.

If you have read all four topics in this introductory series on segment routing, then you are now ready to explore SR-MPLS and SRv6 in detail.

Overview of Segment Routing Statistics

Traffic statistics in a segment routing network can be recorded in an OpenConfig compliant format for the Layer 3 interfaces. The statistics is recorded for the Source Packet Routing in Networking (SPRING) traffic only, excluding RSVP and LDP-signaled traffic, and the family MPLS statistics per interface is accounted for separately. The SR statistics also includes SPRING traffic statistics per link aggregation group (LAG) member, and per segment identifier (SID). To enable recording of segment routing statistics, include `sensor-based-stats` statement at the `[edit protocol isis source-packet-routing]` hierarchy level.

Earlier, sensors were available for collecting segment routing statistics for MPLS transit traffic only, which is MPLS-to-MPLS in nature. On MX Series routers with MPC and MIC interfaces and PTX Series routers, additional sensors are introduced to collect segment routing statistics for MPLS ingress traffic, which is IP-to-MPLS in nature. With this feature, you can enable sensors for label IS-IS segment routing traffic only, and stream the statistics to a gRPC client.

You can enable the segment routing statistics for MPLS ingress traffic using the `egress` option under the `per-sid` configuration statement. The resource name for the per-sid egress functionality is:

`/junos/services/segment-routing/sid/egress/usage/`

You can view the label IS-IS route association with the sensors using the `show isis spring sensor info` command output. This command does not display counter values of the actual sensors.

The segment routing statistics records are exported to a server. You can view segment routing statistics data from the following the OpenConfig paths:

- `/mpls/signalling-protocols/segment-routing/aggregate-sid-counters/aggregate-sid-counter[ip-addr='L-ISIS-10.1.1.1']/state/counters[name='oc-xxx']/out-pkts`
- `/mpls/signalling-protocols/segment-routing/aggregate-sid-counters/aggregate-sid-counter[ip-addr='L-ISIS-10.1.1.1']/state/counters[name='oc-xxx']/out-pkts`

NOTE:

- Graceful Routing Engine switchover (GRES) is not supported for segment routing statistics.

  Nonstop active routing (NSR) is not supported for label IS-IS. During a Routing Engine switchover, a new sensor is created in the new primary Routing Engine, replacing the sensor created by the previous primary Routing Engine. As a result, at the time of a Routing Engine switchover, the segment routing statistics counter start from zero.

- Graceful restart is not supported for label IS-IS.

  In case of graceful restart, the existing sensor is deleted and a new sensor is created during IS-IS initialization. The segment routing statistics counter restarts from zero.

- In-service software upgrade (ISSU) and nonstop software upgrade (NSSU) are not supported. In such cases, the segment routing statistics counter is restarted.

- Zero-statistics segment routing data is suppresses and does not get streamed to the gRPC clients.

RELATED DOCUMENTATION

- sensor-based-stats
- sensor (Junos Telemetry Interface)
- sensor-based-stats (Junos Telemetry Interface)

2

CHAPTER

Segment Routing over MPLS

IN THIS CHAPTER

Segment Routing Basics

IN THIS SECTION

Overview of Adjacency Segments, Anycast Segments, and Configurable SRGB in SPRING

IN THIS SECTION

Segment routing (SR) or Source Packet Routing in Networking (SPRING) is a control-plane architecture that enables an ingress router to steer a packet through a specific set of nodes and links without relying

on the intermediate nodes in the network to determine the actual path it should take. SPRING enables automation of a network by using a software-defined network (SDN) controller for traffic steering and traffic engineering in a WAN packet network. To steer packets through the specified set of nodes and links, the ingress router prepends packets with segments that contain an appropriate combination of tunnels. Each segment is associated with an identifier, which is referred to as the segment identifier (SID). An ordered list of segments is encoded as a stack of labels. Every node in the segment routing domain is allocated labels based on the availability of the dynamic label range. A segment routing global block (SRGB) is the range of label values reserved for segment routing.

Configure the SRGB range label for SPRING. These labels are used by SPRING within the IS-IS or OSPF domain. This way the labels advertised in segment routing are more predictable and deterministic across the segment routing domain. To configure the starting index value of the SRGB label block, configure:

- The `start-label start-label-block-value` statement at the `[edit protocols isis source-packet-routing srgb]` hierarchy level for IS-IS.
- The `start-label start-label-block-value` statement at the `[edit protocols ospf source-packet-routing srgb]` hierarchy level for OSPF.

To configure the index range of the SRGB label block, configure:

- The `index-range` *`value`* statement at the `[edit protocols isis source-packet-routing srgb]` hierarchy level for IS-IS.
- The `index-range` *`value`* statement at the `[edit protocols isis source-packet-routing srgb]` hierarchy level for IS-IS.

Define the SRGB for the IS-IS protocol, and provide prefix anycast segments in addition to node segments to prefixes that are advertised by the IS-IS protocol through policy configuration. Junos OS also extends support to SPRING anycast segments and configurable adjacency segment indexes for the IS-IS protocol.

Benefits of Anycast Segments, Adjacency Segments, and Configurable SRGB

- With the support for anycast prefix segments on Junos OS, you can configure multiple routers to advertise the same prefix with the same SID, which facilitates load balancing.
- Configuring the adjacency hold time helps retain segments for a specified period of time after a link flaps and ensures faster convergence after a link fails.
- Configuring the SRGB label range ensures that the labels are more predictable across segment routing domain.

Configurable Segment Routing Global Block

A segment is encoded as an MPLS label. An ordered list of segments is encoded as a stack of labels. Every node in the segment routing domain is allocated labels by the node label manager based on the index range configured for source packet routing. These labels are allocated to the node segment based on the availability of the dynamic label range managed by node label manager. An SRGB is the range of label values used in segment routing.

Junos OS supports SRGB under `mpls label-range`. You can configure an available SRGB label range for the IS-IS, OSPF, and BGP protocols so that the labels are predictable across segment routing domains. Ensure that the configured SRGB labels are not used by any other application.

To configure SRGB under the global MPLS label range, execute the following command: `set protocols mpls-label-range srgb` *`range-start`* `end` *`range-end`*. You don't need separate label ranges for each protocol because Junos OS centralizes SRGB under MPLS. This enables IS-IS, OSPF, and BGP share one predictable pool, conserving label space

Adjacency Segments and Prefix Segments

A node steers a packet to its destination through an ordered list of instructions, called segments. Essentially, segment routing engages interior gateway protocols (IGPs) such as IS-IS and OSPF to advertise two types of network segments:

- Adjacency segments—A strict forwarded single-hop tunnel that carries packets over a specific link between two nodes, irrespective of the link cost.
- Prefix segments—A multihop tunnel that uses equal cost multi-hop aware shortest path links to reach a prefix. The prefix SID supports both IPv4 and IPv6 prefixes. A node segment is a special case of prefix segment, where every node computes shortest path to the node segment and programs in the forwarding plane. An anycast segment is also a type of prefix segment that identifies a set of routers to advertise the same prefix with the same SID value.

You can redistribute Segment Routing (SR) prefix-Segment Identifiers (SIDs) across OSPF IGP instances using route policy without explicitly specifying a prefix-segment index. This feature standardizes SR labels across instances and improves operational efficiency. Configure a policy with the `from prefix-segment` statement to match routes carrying prefix-segment information. In the `then` clause, use `prefix-segment redistribute` to inherit segment information from the matched route. We also support stitching `mpls.0` routes to enable interoperability between different IGP instances.

Configurable Adjacency Segment Hold Time

The IS-IS protocol creates adjacency segments per adjacency, level, and address family (one each for IPv4 and IPv6). An MPLS label is allocated for each adjacency segment that gets created. These labels are allocated after the adjacency status of the segment changes to the up state. Configure a hold time to

ensure that IS-IS does not release the segments immediately after a link flaps or goes down, but retains them for the configured hold time duration. The default hold time for adjacency segments in IS-IS protocol is 300 seconds.

The OSPF protocol creates adjacency segments per adjacency. To ensure adjacency segments are retained during adjacency or link flaps, the adjacency segments are not released immediately during the link down. The default hold time for adjacency segments in OSPF protocol is 180 seconds.

Prefix Segment Index

Currently, Junos OS enables you to configure a SPRING node SID for IPv4 and IPv6 address families for each routing instance. This node SID is attached to an IPv4 and IPv6 router ID if the router ID is configured on the loopback interface. Otherwise, the lowest IP address assigned to the loopback interface is chosen as the node SID. Configuring a node SID through policy allows you to choose the loopback address that gets the node SID. If the node SID configuration exists and a policy is defined for node SID selection for the same prefix, then the policy configuration takes precedence.

Designate prefix segment indexes to prefix SIDs, both anycast and node SIDs, that are advertised in IS-IS through policy configuration. Remote routers use the SRGB and the index to derive labels for a specific prefix. After the prefix segment indexes are provisioned, the devices running Junos OS advertise them in one or more of the following IS-IS TLV types by using a new Prefix-SID Sub-TLV (type 3):

- IP Prefix TLV (type 135)
- MT IP Prefix TLV (type 235)
- IPV6 Prefix Reachability TLV (type 236)
- MT IPV6 Prefix Reachability TLV (type 237)

You can similarly designate prefix segment indexes to prefix SIDs, both anycast and node SIDs, that are advertised in OSPF through policy configuration. Remote routers use this index to consolidate prefixes into respective SRGBs and to derive the segment identifier and forward the traffic destined for a specific prefix.

Anycast Segments

An IGP anycast segment is an IGP prefix segment that identifies a set of routers. An anycast segment enforces forwarding based on the equal-cost multipath-aware shortest-path toward the closest node of the anycast set. Within an anycast group, all the routers advertise the same prefix with the same SID value, which facilitates load balancing.

Prefix SID redistribution into OSPF through policy configuration

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
25.4R1	Starting in Junos OS and Junos OS Evolved Release 25.4R1, you can redistribute Segment Routing (SR) prefix-Segment Identifiers (SIDs) across OSPF IGP instances using route policy without explicitly specifying a prefix-segment index.

RELATED DOCUMENTATION

Configure Anycast and Prefix segments in SPRING for IS-IS Protocol | 43

Configure Segment Routing Global Blocks Label Ranges in SPRING for IS-IS Protocol | 41

Example: Configuring SRGB in Segment Routing for IS-IS | 32

prefix-segment

srgb

traffic-engineering

Example: Configuring SRGB in Segment Routing for IS-IS

IN THIS SECTION

This example shows how to define the segment routing label block (SRGB) label range for segment packet routing in networking (SPRING) or segment routing (SR) for the IS-IS protocol. This configuration ensures that the labels are more predictable across the segment routing domain with a beneficial impact on network speed.

NOTE: Our content testing team has validated and updated this example.

Requirements

This example uses the following hardware and software components:

- Two MX Series routers
- Junos OS Release 17.2 or later running on all devices
  - Updated and revalidated using vMX on Junos OS Release 21.1R1.

NOTE: Are you interested in getting hands-on experience on this feature?
Visit Juniper vLabs to reserve your pre-configured vLab Sandbox: Segment Routing - Basic and try it out for free!

Before you configure the SRGB label range for segment routing in the IS-IS domain, ensure that you've configured the routing and signaling protocols.

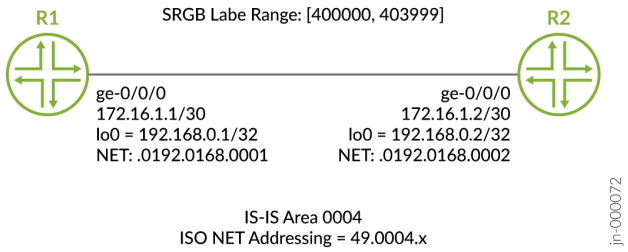
Overview

IN THIS SECTION

Currently, Junos OS allows you to configure only node segment indices. The value of the start label depends on the dynamic label available in the system. Because there is no predictability of the dynamic label range being allocated to the SRGB, Junos OS allows you to configure the SRGB label range used by segment routing. The labels in the SRGB range are used for segment routing in the IS-IS domain. This means the labels advertised are more predictable and deterministic across the segment routing domain.

Topology

Figure 1 shows SRGB configured on router R1 and router R2.



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter `commit` from configuration mode.

R1

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.1/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:1:1::1/128
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set interfaces lo0 unit 0 family iso address 49.0004.0192.0168.0001.00
set interfaces lo0 unit 0 family inet6 address 2001:db8:10:10::1/128
set protocols isis interface ge-0/0/0.0
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 4000
set protocols isis source-packet-routing node-segment ipv4-index 2001
```

```
set protocols isis source-packet-routing node-segment ipv6-index 3001
set protocols isis level 1 disable
set protocols mpls interface ge-0/0/0.0
```

R2

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.2/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:1:1::2/64
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family iso address 49.0004.0192.0168.0002.00
set interfaces lo0 unit 0 family inet6 address 2001:db8:20:20::1/128
set protocols isis interface ge-0/0/0.0
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 4000
set protocols isis source-packet-routing node-segment ipv4-index 2002
set protocols isis source-packet-routing node-segment ipv6-index 3002
set protocols isis level 1 disable
set protocols mpls interface ge-0/0/0.0
```

Configuring Device R1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure device R1:

NOTE: Repeat this procedure for device R2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure enhanced-ip mode on the MX Series because the SRGB functionality is supported on routers with MPCs and MIC interfaces only. A system reboot is required after you commit this configuration.

```
[edit chassis]
user@R1# set network-services enhanced-ip
```

2. Configure the interfaces.

```
[edit interfaces]
user@R1# set ge-0/0/0 unit 0 family inet address 172.16.1.1/30
user@R1# set ge-0/0/0 unit 0 family iso
user@R1# set ge-0/0/0 unit 0 family inet6 address 2001:db8:1:1::1/128
user@R1# set ge-0/0/0 unit 0 family mpls
user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
user@R1# set lo0 unit 0 family iso address 49.0004.0192.0168.0001.00
user@R1# set lo0 unit 0 family inet6 address 2001:db8:10:10::1/128
```

3. Configure the MPLS protocol on the interface. For segment routing to work, you can configure any of the statements under the [`edit protocols mpls`] hierarchy.

```
[edit protocols]
user@R1# set mpls interface ge-0/0/0.0
```

4. Configure the start label and index range of SRGB.

NOTE:

- Ensure that the MPLS label for a binding segment ID (SID) is the sum of the SRGB start label and SID index value. In addition, SID index value must be less than or equal to the index-range value specified in the configuration.
- Junos does not check whether the SID index is within the SRGB's range when the SID index is assigned through an IS-IS export policy. If you configure an index that is out of range of the configured SRGB, you won't see any error message in the logs or while committing the configuration. Junos OS shows a commit error only when you

configure the SID under the **[edit protocols isis source-packet-routing]** hierarchy level.

```
[edit protocols]
user@R1# set isis source-packet-routing srgb start-label 400000
user@R1# set isis source-packet-routing srgb index-range 4000
```

5. Configure the IPv4 index value of the node segment.

```
[edit protocols]
user@R1# set isis source-packet-routing node-segment ipv4-index 2001
```

6. Configure the IPv6 index value of the node segment.

```
[edit protocols]
user@R1# set isis source-packet-routing node-segment ipv6-index 3001
```

7. Disable level 1, configure the IS-IS protocol on the interface, and configure loopback interface lo0.0 as passive..

```
[edit protocols]
user@R1# set isis level 1 disable
user@R1# set isis interface ge-0/0/0.0
user@R1# set isis interface lo0.0 passive
```

Results

From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, and **show protocols** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show chassis
network-services enhanced-ip;
```

```
user@R1# show interfaces
ge-0/0/0 {
```

```
    unit 0 {
        family inet {
            address 172.16.1.1/30;
        }
        family iso;
        family inet6 {
            address 2001:db8:1:1::1/128;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
        family iso {
            address 49.0004.0192.0168.0001.00;
        }
        family inet6 {
            address 2001:db8:10:10::1/128;
        }
    }
}
```

```
user@R1# show protocols
isis {
    interface ge-0/0/0.0;
    interface lo0.0 {
        passive;
    }
    source-packet-routing {
        srgb start-label 400000 index-range 4000;
        node-segment {
            ipv4-index 2001;
            ipv6-index 3001;
        }
    }
    level 1 disable;
}
mpls {
```

```
    interface ge-0/0/0.0;
}
```

Verification

SUMMARY

Confirm that the configuration is working properly.

IN THIS SECTION

Verifying the Configurable SRGB

Purpose

Verify the configurable SRGB label range in the IS-IS overview information.

Action

From operational mode, run the `show isis overview` command to display the IS-IS overview information.

```
user@R1> show isis overview
Instance: master
  Router ID: 128.53.50.230
  IPv6 Router ID: abcd::128:53:50:230
  Hostname: R1
  Sysid: 1280.5305.0230
  Areaid: 47.0005.80ff.f800.0000.0108.0001
  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
  Source Packet Routing (SPRING): Enabled
```

```
    SRGB Config Range :
      SRGB Start-Label : 400000, SRGB Index-Range : 4000
    SRGB Block Allocation: Success
      SRGB Start Index : 400000, SRGB Size : 4000, Label-Range: [ 400000, 403999 ]
    Node Segments: Enabled
      Ipv4 Index : 2001, Ipv6 Index : 3001
    SRv6: Disabled
  Post Convergence Backup: Disabled
  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, Narrow metrics are enabled
    Source Packet Routing is enabled
```

Meaning

The output displays the configured SRGB start label and the SRGB index range. The end of the SRGB label range is the summation of the start label value and the index range. All devices in the segment routing domain must have the same SRGB range values.

RELATED DOCUMENTATION

> *Overview of Adjacency Segments, Anycast Segments, and Configurable SRGB in SPRING*
>
> *Configuring Segment Routing Global Blocks Label Ranges in SPRING for IS-IS Protocol*
>
> *source-packet-routing*
>
> vLab Sandbox: Segment Routing - Basic

Configure Segment Routing Global Blocks Label Ranges in SPRING for IS-IS Protocol

Segment routing (SR) or source packet routing in networking (SPRING) is a control-plane architecture that enables an ingress router to steer a packet through a specific set of nodes and links without relying on the intermediate nodes in the network to determine the actual path it should take. The label range for a segment routing global block (SRGB) is the range of label values used in segment routing. You can configure the start of the label range and the index range. The end of the label range is the summation of the start label value and the index range.

Before you configure SPRING SRGB for IS-IS protocol, you must:

- Configure the router interfaces.
- Configure ISIS.

To configure SPRING SRGB label range on a device:

1. Configure the start-label and index-range of SRGB. The start label value indicates the start of the SPRING label block and the index range along with the start label indicate the end of the label block.

NOTE:

- Ensure that the MPLS label for a binding segment ID (SID) is the sum of the SRGB start label and SID index value. In addition, SID index value must be less than or equal to the index-range value specified in the configuration.
- Junos OS does not check whether the SID index is within the SRGB's range when the SID index is assigned through an ISIS export policy. If you configure an index that is out of range of the configured SRGB, you won't see any error message in the logs or while committing the configuration. Junos OS shows a commit error only when you configure the SID under the **[edit protocols isis source-packet-routing]** hierarchy level.

```
[edit protocols isis source-packet-routing]
user@host# set srgb start-label start-label-value
user@host# set srgb index-range index-range-value
```

NOTE: The default value for the index range is 4096. This causes chunks of 256 label blocks being dynamically allocated by the label manager depending on the availability.

For example, configure SRGB with start-label 800,000 and index-range 40,000. The start label of the SPRING label block is 800,000 and the end of the label block is 840,000.

```
[edit protocols isis source-packet-routing]
user@host# set srgb start-label 800000
user@host# set srgb index-range 40000
```

NOTE: Ensure that the labels in the SRGB label range are not used by any other applications. If a label in the configured label range is used by another application, then a syslog error message `RPD_ISIS_SRGBALLOCATIONFAIL` is logged to indicate that the label manager is unable to allocate the requested SRGB label range. To free up the configured label range, check the label ranges configured at the `[edit protocol mpls label-range]` hierarchy level and re-configure the SRGB label range with a label range that is available and restart the routing protocol process (RPD).

2. Configure the value of IPv4 node segment index.

```
[edit protocols isis source-packet-routing]
user@host# set node-segment ipv4-index ipv4-index-value
```

For example, configure 1001 for IPv4 node segment index.

```
[edit protocols isis source-packet-routing]
user@host# set node-segment ipv4-index 1001
```

3. Configure the value of IPv6 node segment index.

```
[edit protocols isis source-packet-routing]
user@host# set node-segment ipv6-index ipv6-index-value
```

For example, configure 2001 for IPv6 node segment index.

```
[edit protocols isis source-packet-routing]
user@host# set node-segment ipv6-index 2001
```

RELATED DOCUMENTATION

Understanding Adjacency Segments, Anycast Segments, and Configurable SRGB in SPRING

Example: Configuring Segment Routing Global Blocks in SPRING for IS-IS

source-packet-routing

Configure Anycast and Prefix segments in SPRING for IS-IS Protocol

Segment routing (SR) or source packet routing in networking (SPRING) is a control-plane architecture that enables an ingress router to steer a packet through a specific set of nodes and links without relying on the intermediate nodes in the network to determine the actual path it should take. Segment routing global block (SRGB) is the range of label values used in segment routing. Junos OS allows you to configure prefix segment identifier (SID) and node SID to prefixes that are advertised in IS-IS through policy configuration.

Before you configure SPRING SRGB, prefix SID, and anycast SID for the IS-IS protocol, you must:

- Configure the router interfaces.
- Configure the router ID.
- Configure IS-IS.

To configure device R1 with SPRING SRGB, prefix SID, and anycast SID for IS-IS protocols:

1. Configure the start-label and index-range of SRGB.

```
[edit protocols isis source-packet-routing]
user@host# set srgb start-label start-label-value
user@host# set srgb index-range index-range-value
```

For example, configure SRGB with start-label 800000 and index-range 40000 .

```
[edit protocols isis source-packet-routing]
user@host# set srgb start-label 800000
user@host# set srgb index-range 40000
```

2. Configure the routing policy to match a route (IPv4 or IPv6) exactly. Configure the index and the node segment of the prefix segment for a given term and accept the routing policy.

```
[edit policy-options policy-statement policy-name term term-value]
user@host#  set from route-filter IP address exact
user@host#  set then prefix-segment index index-value
user@host#  set then prefix-segment node-segment
user@host#  set accept
```

NOTE: Configure node segment as /32 prefix on loopback interface (lo0.0) or on a valid stub interface.

For example, configure the routing policy to match the IPv4 route exactly. Configure the index and the node segment of the prefix segment for a given term and accept the routing policy.

```
[edit policy-options policy-statement policy-name term term-value]
user@host#  set from route-filter 198.51.100.1/32 exact
user@host#  set then prefix-segment index index-value
user@host#  set then prefix-segment node-segment
user@host#  set accept
```

For example, configure the routing policy to match the IPv6 route exactly. Configure the index and the node segment of the prefix segment for a given term and accept the routing policy.

```
[edit policy-options policy-statement policy-name term term-value]
user@host#  set from route-filter 2001:db8::/32 exact
user@host#  set then prefix-segment index index-value
user@host#  set then prefix-segment node-segment
user@host#  set accept
```

3. Configure the index and the node segment of the prefix segment for a given term and accept the routing policy.

```
[edit policy-options policy-statement policy-name term term-value then]
user@host# set prefix-segment index index-value
user@host# set prefix-segment node-segment
user@host#  set accept
```

For example, configure the prefix segment with index 1004 and the node segment for term 1 of policy statement prefix SID and accept the routing policy.

```
[edit policy-options policy-statement prefix-sid term 1 then]
user@host# set prefix-segment index 1004
user@host# set prefix-segment node-segment
user@host#  set accept
```

4. Configure the routing policy with the same prefix (IPv4 or IPv6)and same prefix segment on more than one routers for anycast SID.

NOTE: For anycast prefix SID, configure prefix SID on loopback interface(lo0.0).

```
[edit policy-options policy-statement prefix-sid term 1 ]
user@host# set from route-filter IP address exact
user@host# set then prefix-segment index index-value
user@host# set then accept
```

For example, configure IPv4 prefix 198.51.100.1/32 with prefix segment 1000 on two routers R0 and R1 for anycast SID.

```
[edit policy-options policy-statement prefix-sid term 1 ]
user@host#  set from route-filter 198.51.100.1/32 exact
user@host#  set then prefix-segment index 1000
user@host# set then accept
```

For example, configure IPv6 prefix 2001:db8::/32 with prefix segment 1000 on two routers R0 and R1 for anycast SID.

```
[edit policy-options policy-statement prefix-sid term 1 ]
user@host#  set from route-filter 2001:db8::/32 exact
user@host#  set then prefix-segment index 2000
user@host#  set then accept
```

5. Configure export policy on the IS-IS protocol.

```
[edit protocols isis]
user@host#  export prefix-sid
```

6. Configure explicit NULL to enable E and P bits in all prefix SID advertisements.

```
[edit protocol isis source-packet-routing]
user@host# set explicit-null
```

7. Configure adjacency segment hold time to retain segment adjacency.

```
[edit protocol isis source-packet-routing]
user@host# set adjacency-segment hold-time hold-time
```

For example, configure adjacency segments with 240,000 milliseconds hold time.

```
[edit protocol isis source-packet-routing]
user@host# set adjacency-segment hold-time 240000
```

RELATED DOCUMENTATION

Understanding Adjacency Segments, Anycast Segments, and Configurable SRGB in SPRING

source-packet-routing

traffic-engineering

Example: Configuring Anycast and Prefix Segments in SPRING for IS-IS

IN THIS SECTION

This example shows how to configure prefix segments, segment-routing global blocks (SRGBs), adjacency segments hold time, and explicit null flag for prefix segments in source packet routing in networking (SPRING) or segment routing (SR). This configuration helps in simplifying the network thereby increasing the speed of the network.

Requirements

This example uses the following hardware and software components:

- Eight MX Series routers.
- Junos OS Release 17.2 or later running on all devices.

Before you configure prefix segments in SPRING, be sure you configure routing and signaling protocols.

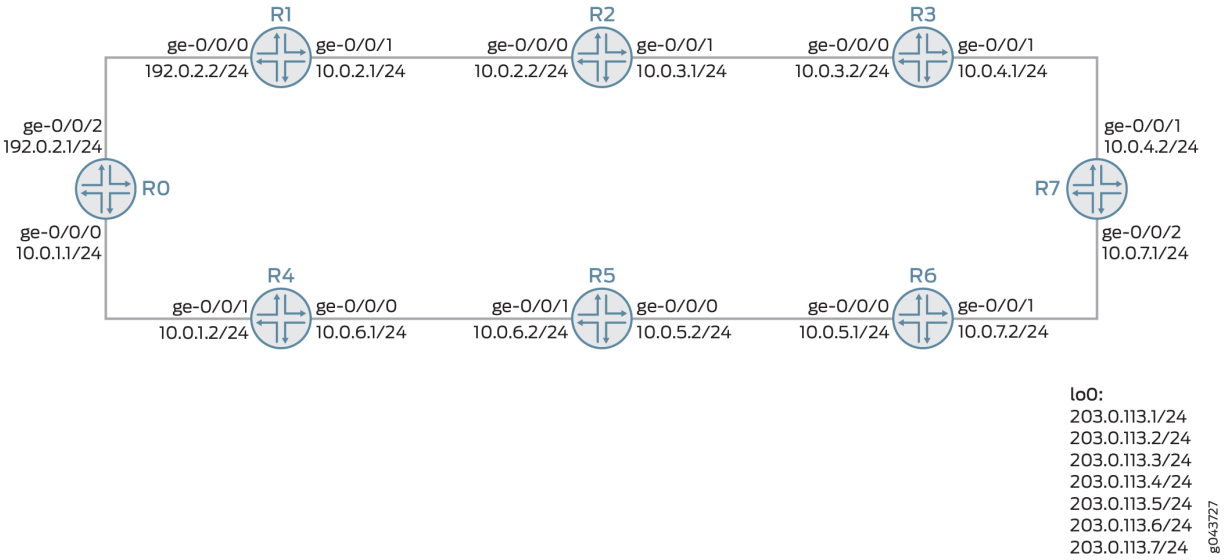
Overview

IN THIS SECTION

You can provide prefix segment identifier (SID) and node SID to prefixes that are advertised in IS-IS by configuring policies. Prefix segment index is the index assigned to a specific prefix. This is used by all other remote routers in the network to index the prefix into respective segment-routing global blocks (SRGBs) to derive the segment identifier and to forward the traffic destined for this prefix. The prefix SID supports both IPv4 and IPv6 prefixes. The IS-IS protocol creates adjacency segments per adjacency, level, and address family (one each for IPv4 and IPv6).

Topology

Figure 1 shows SRGBs, prefix segments, and adjacency hold time configured in SPRING on routers R0 to R7.



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter `commit` from configuration mode.

NOTE: This topology demonstrates IPv4 prefixes. The same is applicable for IPv6 prefixes.

R0

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
```

```
set interfaces ge-0/0/0 unit 1 family inet address 10.0.1.1/24
set interfaces ge-0/0/0 unit 1 family iso
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 1 vlan-id 1
set interfaces ge-0/0/2 unit 1 family inet address 192.10.12.1/24
set interfaces ge-0/0/2 unit 1 family iso
set interfaces ge-0/0/2 unit 1 family mpls maximum-labels 5
set interfaces lo0 unit 0 family inet address 203.0.113.1/24
set interfaces lo0 unit 0 family iso address 49.0001.000a.0a0a.0a00
set routing-options autonomous-system 100
set routing-options router-id 203.0.113.1
set routing-options forwarding-table export pplb
set routing-options forwarding-table chained-composite-next-hop ingress l3vpn
set protocols mpls traffic-engineering
set protocols mpls interface all
set protocols isis export prefix-sid
set protocols isis backup-spf-options remote-backup-calculation
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis graceful-restart restart-duration 30
set protocols isis source-packet-routing adjacency-segment hold-time 240000
set protocols isis source-packet-routing srgb start-label 800000
set protocols isis source-packet-routing srgb index-range 40000
set protocols isis source-packet-routing explicit-null



set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 203.0.113.1/24 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1000
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
```

R1

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
set interfaces ge-0/0/0 unit 1 family inet address 192.0.2.2/24
set interfaces ge-0/0/0 unit 1 family iso
set interfaces ge-0/0/0 unit 1 family mpls
```

```
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 1 family inet address 10.0.2.1/24
set interfaces ge-0/0/1 unit 1 family iso
set interfaces ge-0/0/1 unit 1 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.2/24
set interfaces lo0 unit 0 family iso address 49.0001.0001.0101.0100
set routing-options router-id 203.0.113.2
set routing-options forwarding-table export pplb
set protocols mpls traffic-engineering
set protocols mpls interface all
set protocols isis export prefix-sid
set protocols isis backup-spf-options per-prefix-calculation
set protocols isis backup-spf-options remote-backup-calculation
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering family inet shortcuts
set protocols isis graceful-restart restart-duration 30
set protocols isis source-packet-routing adjacency-segment hold-time 240000
set protocols isis source-packet-routing srgb start-label 800000
set protocols isis source-packet-routing srgb index-range 40000
set protocols isis source-packet-routing explicit-null



set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis label-switched-path to_r2
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 203.0.113.2/24 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1001
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set policy-options policy-statement setpref from protocol isis
set policy-options policy-statement setpref from level 2
set policy-options policy-statement setpref then preference 11
set policy-options policy-statement setpref then local-preference 11
set policy-options policy-statement setpref then accept
```

R2

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 vlan-tagging
```

```
set interfaces ge-0/0/0 encapsulation flexible-ethernet-services
set interfaces ge-0/0/0 unit 1 vlan-id 1
set interfaces ge-0/0/0 unit 1 family inet address 10.0.2.2/24
set interfaces ge-0/0/0 unit 1 family iso
set interfaces ge-0/0/0 unit 1 family mpls
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 encapsulation flexible-ethernet-services
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 1 family inet address 10.0.3.1/24
set interfaces ge-0/0/1 unit 1 family iso
set interfaces ge-0/0/1 unit 1 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.3/24
set interfaces lo0 unit 0 family iso address 49.0001.0002.0202.0200
set routing-options router-id 203.0.113.3
set routing-options forwarding-table export pplb
set protocols mpls interface all
set protocols isis export leakl2tol1
set protocols isis export prefix-sid
set protocols isis backup-spf-options remote-backup-calculation
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis source-packet-routing adjacency-segment hold-time 240000
set protocols isis source-packet-routing srgb start-label 800000
set protocols isis source-packet-routing srgb index-range 40000
set protocols isis source-packet-routing explicit-null



set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis label-switched-path to_r1
set policy-options policy-statement leakl2tol1 from protocol isis
set policy-options policy-statement leakl2tol1 from level 2
set policy-options policy-statement leakl2tol1 to protocol isis
set policy-options policy-statement leakl2tol1 to level 1
set policy-options policy-statement leakl2tol1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 203.0.113.3/24 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1002
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
```

R3

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
set interfaces ge-0/0/0 unit 1 family inet address 10-.0.3.2/24
set interfaces ge-0/0/0 unit 1 family iso
set interfaces ge-0/0/0 unit 1 family mpls
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 1 family inet address 10.0.4.1/24
set interfaces ge-0/0/1 unit 1 family iso
set interfaces ge-0/0/1 unit 1 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.4/24
set interfaces lo0 unit 0 family iso address 49.0001.0003.0303.0300
set routing-options router-id 203.0.113.4
set routing-options forwarding-table export pplb
set protocols mpls interface all
set protocols isis export prefix-sid
set protocols isis backup-spf-options remote-backup-calculation
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis source-packet-routing adjacency-segment hold-time 240000
set protocols isis source-packet-routing srgb start-label 800000
set protocols isis source-packet-routing srgb index-range 40000
set protocols isis source-packet-routing explicit-null



set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 203.0.113.4/24 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1003
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
```

R4

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
```

```
set interfaces ge-0/0/0 unit 1 family inet address 10.0.6.2/24
set interfaces ge-0/0/0 unit 1 family iso
set interfaces ge-0/0/0 unit 1 family mpls
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 1 family inet address 10.0.1.2/24
set interfaces ge-0/0/1 unit 1 family iso
set interfaces ge-0/0/1 unit 1 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.5/24
set interfaces lo0 unit 0 family iso address 49.0001.0004.0404.0400
set routing-options router-id 203.0.113.5
set routing-options forwarding-table export pplb

set protocols mpls interface all
set protocols isis export prefix-sid
set protocols isis backup-spf-options remote-backup-calculation
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis source-packet-routing adjacency-segment hold-time 240000
set protocols isis source-packet-routing srgb start-label 800000
set protocols isis source-packet-routing srgb index-range 40000
set protocols isis source-packet-routing explicit-null



set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 203.0.113.5/24 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1004
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
```

R5

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
set interfaces ge-0/0/0 unit 1 family inet address 10.0.5.2/24
set interfaces ge-0/0/0 unit 1 family iso
set interfaces ge-0/0/0 unit 1 family mpls
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
```

```
set interfaces ge-0/0/1 unit 1 family inet address 10.0.6.2/24
set interfaces ge-0/0/1 unit 1 family iso
set interfaces ge-0/0/1 unit 1 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.6/24
set interfaces lo0 unit 0 family iso address 49.0001.0005.0505.0500
set routing-options router-id 203.0.113.6
set routing-options forwarding-table export pplb
set protocols mpls interface all
set protocols isis export leakl2tol1
set protocols isis export prefix-sid
set protocols isis backup-spf-options remote-backup-calculation
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis source-packet-routing adjacency-segment hold-time 240000
set protocols isis source-packet-routing srgb start-label 800000
set protocols isis source-packet-routing srgb index-range 40000
set protocols isis source-packet-routing explicit-null



set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set policy-options policy-statement leakl2tol1 from protocol isis
set policy-options policy-statement leakl2tol1 from level 2
set policy-options policy-statement leakl2tol1 to protocol isis
set policy-options policy-statement leakl2tol1 to level 1
set policy-options policy-statement leakl2tol1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 203.0.113.6/24 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1005
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
```

R6

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
set interfaces ge-0/0/0 unit 1 family inet address 10.0.5.1/24
set interfaces ge-0/0/0 unit 1 family iso
set interfaces ge-0/0/0 unit 1 family mpls
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
```

```
set interfaces ge-0/0/1 unit 1 family inet address 10.0.6.2/24
set interfaces ge-0/0/1 unit 1 family iso
set interfaces ge-0/0/1 unit 1 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.7/24
set interfaces lo0 unit 0 family iso address 49.0001.0006.0606.0600
set routing-options router-id 203.0.113.7
set routing-options forwarding-table export pplb
set protocols mpls interface all
set protocols isis export prefix-sid
set protocols isis backup-spf-options remote-backup-calculation
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis source-packet-routing adjacency-segment hold-time 240000
set protocols isis source-packet-routing srgb start-label 800000
set protocols isis source-packet-routing srgb index-range 40000
set protocols isis source-packet-routing explicit-null



set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 203.0.113.7/24 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1006
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
```

R7

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 1 family inet address 10.0.4.2/24
set interfaces ge-0/0/1 unit 1 family iso
set interfaces ge-0/0/1 unit 1 family mpls
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 1 vlan-id 1
set interfaces ge-0/0/2 unit 1 family inet address 10.0.7.1/24
set interfaces ge-0/0/2 unit 1 family iso
set interfaces ge-0/0/2 unit 1 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.8/24
set interfaces lo0 unit 0 family iso address 49.0001.0007.0707.0700
set routing-options router-id 203.0.113.8
```

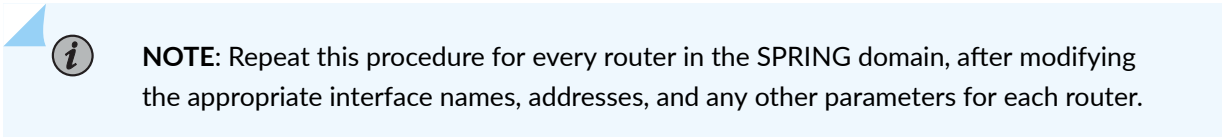
```
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols mpls explicit-null
set protocols mpls interface all
set protocols isis export prefix-sid
set protocols isis backup-spf-options remote-backup-calculation
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis source-packet-routing adjacency-segment hold-time 240000
set protocols isis source-packet-routing srgb start-label 800000
set protocols isis source-packet-routing srgb index-range 40000
set protocols isis source-packet-routing explicit-null



set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 203.0.113.8/24 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1007
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set policy-options policy-statement setpref from protocol isis
set policy-options policy-statement setpref from level 2
set policy-options policy-statement setpref then preference 11
set policy-options policy-statement setpref then local-preference 11
set policy-options policy-statement setpref then accept
set policy-options policy-statement stat term 1 from protocol static
set policy-options policy-statement stat term 1 then accept
```

Configuring Router R4

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure Router R4:



NOTE: Repeat this procedure for every router in the SPRING domain, after modifying the appropriate interface names, addresses, and any other parameters for each router.

1. Configure enhanced IP mode on the MX Series router because the SRGB functionality is supported on routers with MPCs and MIC interfaces only. A system reboot is required after you commit this configuration.

```
[edit chassis]
user@R4# set network-services enhanced-ip
```

2. Configure the interfaces.

```
[edit interfaces]
user@R4# set ge-0/0/0 vlan-tagging
user@R4# set ge-0/0/0 unit 1 vlan-id 1
user@R4# set ge-0/0/0 unit 1 family inet address 10.0.6.2/24
user@R4# set ge-0/0/0 unit 1 family iso
user@R4# set ge-0/0/0 unit 1 family mpls
user@R4# set ge-0/0/1 vlan-tagging
user@R4# set ge-0/0/1 unit 1 vlan-id 1
user@R4# set ge-0/0/1 unit 1 family inet address 10.0.1.2/24
user@R4# set ge-0/0/1 unit 1 family iso
user@R4# set ge-0/0/1 unit 1 family mpls
user@R4# set lo0 unit 0 family inet address 203.0.113.5/24
user@R4# set lo0 unit 0 family iso address 49.0001.0004.0404.0400
```

3. Configure the router ID for a routing option.

```
[edit routing-options]
user@R4# set router-id 203.0.113.5
```

4. Configure the export policy for the forwarding table.

```
[edit routing-options]
user@R4# set forwarding-table export pplb
```

5. Configure the MPLS interface.

```
[edit protocols mpls]
user@R4# set interface all
```

6. Configure the export policy for the IS-IS protocol.

```
[edit protocols isis]
user@R4# set export prefix-sid
```

7. Configure backup shortest-path-first options to calculate remote loop-free alternate (LFA) backup next hops and to use SPRING routed paths for protection for the IS-IS protocol.

```
[edit protocols isis]
user@R4# set backup-spf-options remote-backup-calculation
user@R4# set backup-spf-options use-source-packet-routing
```

8. Configure adjacency segment hold time in SPRING for the IS-IS protocol.

```
[edit protocols isis]
user@R4# set source-packet-routing adjacency-segment hold-time 240000
```

9. Configure the start label and index range for segment routing global blocks (SRGBs) in SPRING for the IS-IS protocol.

```
[edit protocols isis]
user@R4# set source-packet-routing srgb start-label 800000
user@R4# set source-packet-routing srgb index-range 40000
```

10. Configure explicit null in SPRING for the IS-IS protocol.

```
[edit protocols isis]
user@R4# set source-packet-routing explicit-null
```

11. Disable the management interface and configure the loopback address as passive for the IS-IS protocol.

```
[edit protocols isis]
user@R4# set interface fxp0.0 disable
user@R4# set interface lo0.0 passive
```

12. Configure per packet load balancing for the routing policy.

```
[edit policy-options policy-statement pplb]
user@R4# set then load-balance per-packet
```

13. Configure the route filter for the routing policy term.

```
[edit policy-options policy-statement prefix-sid]
user@R4# set term 1 from route-filter 203.0.113.5/24  exact
```

14. Configure the index and node segment of the prefix segment for the routing policy term.

```
[edit policy-options policy-statement prefix-sid]
user@R4# set term 1 then prefix-segment index 1004
user@R4# set term 1 then prefix-segment node-segment
user@R4# set term 1 then accept
```

Results

From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R4# show chassis
network-services enhanced-ip;
```

```
user@R4# show interfaces
ge-0/0/0 {
    vlan-tagging;
```

```
    unit 1 {
        vlan-id 1;
        family inet {
            address 10.0.6.2/24;
        }
        family iso;
    }
}
ge-0/0/1 {
    vlan-tagging;
    unit 1{
        vlan-id 1;
        family inet {
            address 10.0.1.2/24;
        }
        family iso;
        family mpls ;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 203.0.113.5/24;
        }
        family iso {
            address 49.0001.0004.0404.0400;
        }
    }
}
```

```
user@R4# show protocols
mpls {
    interface all;
}
isis {
    export prefix-sid;
    backup-spf-options {
        remote-backup-calculation;
        use-source-packet-routing;
    }
    source-packet-routing {
```

```
        adjacency-segment hold-time 240000;
        srgb start-label 800000 index-range 40000;
        explicit-null;
    }
        interface fxp0.0 {
        disable;
    }
    interface lo0.0 {
        passive;
    }
}
```

```
user@R4# show policy-options
policy-statement pplb {
    then {
        load-balance per-packet;
    }
}
policy-statement prefix-sid {
    term 1 {
        from {
            route-filter 203.0.113.5/24  exact;
        }
        then {
            prefix-segment index 1004 node-segment;
            accept;
        }
    }
}
```

```
user@R4# show routing-options
router-id 203.0.113.5;
forwarding-table {
    export pplb;
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the IS-IS Adjacency Routes

Purpose

Verify the adjacency of Router R4.

Action

From operational mode, enter the `show isis adjacency detail` command.

```
user@R4> show isis adjacency detail
R5
  Interface: ge-0/0/0.0, Level: 1, State: Up, Expires in 25 secs
  Priority: 64, Up/Down transitions: 1, Last transition: 1d 23:55:22 ago
  Circuit type: 3, Speaks: IP, IPv6, MAC address: 0:5:86:e:2b:0
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  LAN id: R5.02, IP addresses: 10.0.6.2
  Level 1 IPv4 Adj-SID: 16

R5
  Interface: ge-0/0/0.0, Level: 2, State: Up, Expires in 25 secs
  Priority: 64, Up/Down transitions: 1, Last transition: 1d 23:55:22 ago
  Circuit type: 3, Speaks: IP, IPv6, MAC address: 0:5:86:e:2b:0
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  LAN id: R5.02, IP addresses: 10.0.6.2
```

```
  Level 2 IPv4 Adj-SID: 17

R0
  Interface: ge-0/0/1.0, Level: 1, State: Up, Expires in 7 secs
  Priority: 64, Up/Down transitions: 1, Last transition: 1d 23:49:06 ago
  Circuit type: 3, Speaks: IP, IPv6, MAC address: 0:5:86:5e:8e:1
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  LAN id: R1.02, IP addresses: 10.0.1.1
  Level 1 IPv4 Adj-SID: 18

R0
  Interface: ge-0/0/1.0, Level: 2, State: Up, Expires in 8 secs
  Priority: 64, Up/Down transitions: 1, Last transition: 1d 23:49:06 ago
  Circuit type: 3, Speaks: IP, IPv6, MAC address: 0:5:86:5e:8e:1
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  LAN id: R1.02, IP addresses: 10.0.1.1
  Level 2 IPv4 Adj-SID: 19
```

Meaning

The output shows the IS-IS adjacency details of Router R4 with Router R0 and R5.

Verifying the IS-IS Overview Information

Purpose

Verify the IS-IS overview information of Router R4.

Action

From operational mode, enter the `show isis overview` command.

```
user@R4> show isis overview
Instance: master
  Router ID: 203.0.113.5
  Hostname: R4
  Sysid: 0100.0404.0404
  Areaid: 47.0005.80ff.f800.0000.0108.0001
  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
Restart: Disabled
  Helper mode: Enabled
Layer2-map: Disabled
Source Packet Routing (SPRING): Enabled
  SRGB Config Range:
    SRGB Start-Label : 800000, SRGB Index-Range : 40000
  SRGB Block Allocation: Success
    SRGB Start Index : 800000, SRGB Size : 40000, Label-Range: [ 800000, 839999 ]
  Node Segments: Disabled
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, Narrow metrics are enabled
  Source Packet Routing is enabled
```

Meaning

The output displays the IS-IS overview information of the routing instance along with the SPRING details of Router R4.

Verifying the Segment Routing Route Entries for the IS-IS Protocol

Purpose

Verify the segment routing route entries of the routing table inet.3 for the IS-IS protocol.

Action

From operational mode, enter the `show route table inet.3 protocol isis` command.

```
user@R4> show route table inet.3 protocol isis
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

203.0.113.0/24     *[L-ISIS/14] 00:09:31, metric 10
                       to 10.0.6.2 via ge-0/0/0.0, Push 0
                    > to 10.0.1.1 via ge-0/0/1.0, Push 0
203.0.113.2/32     *[L-ISIS/14] 00:02:44, metric 20
                    > to 10.0.1.1 via ge-0/0/1.0, Push 801001
```

Meaning

The output shows the segment routing routes of routing table inet.3 for the IS-IS protocol.

Verifying the MPLS Segment Routing Route Entries for the IS-IS Protocol

Purpose

Verify the MPLS segment routing route entries for the IS-IS protocol.

Action

From operational mode, enter the `show route table mpls.0 protocol isis` command.

```
user@R4> show route table mpls.0 protocol isis

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

0                  *[MPLS/0] 2d 01:56:20, metric 1
                       to table inet.0
0(S=0)             *[MPLS/0] 2d 01:56:20, metric 1
                       to table mpls.0
1                  *[MPLS/0] 2d 01:56:20, metric 1
                       Receive
2                  *[MPLS/0] 2d 01:56:20, metric 1
```

```
                    to table inet6.0
2(S=0)              *[MPLS/0] 2d 01:56:20, metric 1
                    to table mpls.0
13                  *[MPLS/0] 2d 01:56:20, metric 1
                    Receive
16                  *[L-ISIS/14] 2d 01:52:56, metric 0
                     > to 10.0.6.2 via ge-0/0/0.0, Pop
16(S=0)             *[L-ISIS/14] 00:01:34, metric 0
                     > to 10.0.6.2 via ge-0/0/0.0, Pop
17                  *[L-ISIS/14] 2d 01:52:56, metric 0
                     > to 10.0.6.2 via ge-0/0/0.0, Pop
17(S=0)             *[L-ISIS/14] 00:10:49, metric 0
                     > to 10.0.6.2 via ge-0/0/0.0, Pop
18                  *[L-ISIS/14] 2d 01:46:40, metric 0
                     > to 10.0.1.1 via ge-0/0/1.0, Pop
18(S=0)             *[L-ISIS/14] 00:01:34, metric 0
                     > to 10.0.1.1 via ge-0/0/1.0, Pop
19                  *[L-ISIS/14] 2d 01:46:40, metric 0
                     > to 10.0.1.1 via ge-0/0/1.0, Pop
19(S=0)             *[L-ISIS/14] 00:10:49, metric 0
                     > to 10.0.1.1 via ge-0/0/1.0, Pop
801000              *[L-ISIS/14] 2d 01:46:40, metric 10
                    to 10.0.6.2 via ge-0/0/0.0, Swap 801000
                     > to 10.0.1.1 via ge-0/0/1.0, Swap 0
801000(S=0)         *[L-ISIS/14] 00:01:34, metric 10
                    to 10.0.6.2 via ge-0/0/0.0, Swap 801000
                     > to 10.0.1.1 via ge-0/0/1.0, Pop
801001              *[L-ISIS/14] 2d 01:46:14, metric 10
                    to 10.0.6.2 via ge-0/0/0.0, Swap 801001
                     > to 10.0.1.1 via ge-0/0/1.0, Swap 801001
801002              *[L-ISIS/14] 1d 21:57:31, metric 10
                    to 10.0.6.2 via ge-0/0/0.0, Swap 801002
                     > to 10.0.1.1 via ge-0/0/1.0, Swap 801002
801003              *[L-ISIS/14] 1d 21:56:57, metric 10
                    to 10.0.6.2 via ge-0/0/0.0, Swap 801003
                     > to 10.0.1.1 via ge-0/0/1.0, Swap 801003
801005              *[L-ISIS/14] 2d 01:46:40, metric 10
                    to 10.0.6.2 via ge-0/0/0.0, Swap 0
                     > to 10.0.1.1 via ge-0/0/1.0, Swap 801005
801005(S=0)         *[L-ISIS/14] 00:01:34, metric 10
                    to 10.0.6.2 via ge-0/0/0.0, Pop
                     > to 10.0.1.1 via ge-0/0/1.0, Swap 801005
801006              *[L-ISIS/14] 2d 01:46:40, metric 10
```

```
                    to 10.0.6.2 via ge-0/0/0.0, Swap 801006
                  > to 10.0.1.1 via ge-0/0/1.0, Swap 801006
801007            *[L-ISIS/14] 1d 21:56:24, metric 10
                    to 10.0.6.2 via ge-0/0/0.0, Swap 801007
                  > to 10.0.1.1 via ge-0/0/1.0, Swap 801007
```

Meaning

The output shows the MPLS segment routing route entries for protocol IS-IS.

RELATED DOCUMENTATION

- *Configuring Anycast and Prefix segments in SPRING for IS-IS Protocol*
- *Configuring Segment Routing Global Blocks Label Ranges in SPRING for IS-IS Protocol*
- *Example: Configuring Segment Routing Global Blocks in SPRING for IS-IS*
- *Understanding Adjacency Segments, Anycast Segments, and Configurable SRGB in SPRING*
- *prefix-segment*
- *source-packet-routing*
- *srgb*
- *traffic-engineering*

Static Adjacency Segment Identifier for IS-IS and OSPF

Adjacency segment is a strict forwarded single-hop tunnel that carries packets over a specific link between two nodes, irrespective of the link cost. You can configure static adjacency segment identifier (SID) labels for an interface or an interface group.

Configuring a static adjacency SID on an interface causes the existing dynamically allocated adjacency SID to be removed along with the transit route for the same.

For static adjacency SIDs, the labels are picked from either a static reserved label pool or from the segment routing global block (SRGB).

You can reserve a label range to be used for static allocation of labels using the following configuration:

```
user@host# set protocols mpls label-range static-label-range start-value end-value
```

The static pool can be used by any protocol to allocate a label in this range. You need to ensure that no two protocols use the same static label. The adjacency SIDs can be allocated from this label block

through the configuration using keyword `label`. The `label` value for the specific adjacency SIDs need to be explicitly configured. The specific label is advertised as the adjacency SIDs for that interface for the specific level and address family.

The following is a sample configuration for IS-IS:

```
user@host# set protocols mpls label-range static-label-range 700000 799999;
user@host# set protocols isis source-packet-routing srgb start-label 800000 index-range 4000;
user@host# set protocols isis interface ge-0/0/0.1 level 1 ipv4-adjacency-segment unprotected label 700001;
```

The following is a sample configuration for OSPF:

```
user@host# set protocols mpls label-range static-label-range 700000 799999;
user@host# set protocols ospf source-packet-routing srgb start-label 800000 index-range 4000;
user@host# set protocols ospf area 0.0.0.0 interface ge-0/0/0.1 ipv4-adjacency-segment unprotected label
700001;
```

NOTE: When you use `ipv4-adjacency-segment` command, the underlying interface must be point-to-point.

SRGB is a global label space that is allocated for the protocol based on configuration. The labels in the entire SRGB is available for ISIS to use and are not allocated to other applications/protocols. Prefix SIDs (and Node SIDs) are indexed from this SRGB.

The Adj-SIDs can be allocated from the SRGB using keyword 'index' in the configuration. In such cases, it should be ensured that the Adj-SID index does not conflict with any other prefix SID in the domain. Like Prefix-SIDs, Adj-SIDs will also be configured by mentioning the index with respect to the SRGB. However, the Adj-SID subtlv will still have the SID as a value and the L and V flags are set.

The following is a sample configuration for IS-IS:

```
user@host# set protocols isis source-packet-routing srgb start-label 800000 index-range 4000;
user@host# set protocols isis interface ge-0/0/0.1 level 1 ipv4-adjacency-segment unprotected index 1;
```

Static adjacency SIDs can be configured per address family and also based on whether the protection is required or not. Adjacency SIDs should be configured per level per interface at the [`edit protocols isis interface` *`interface-name`* `level` *`level-num`*] hierarchy level.

The following is a sample configuration for OSPF:

```
user@host# set protocols ospf source-packet-routing srgb start-label 800000 index-range 4000;
user@host# set protocols ospf area 0.0.0.0 interface ge-0/0/0.1 ipv4-adjacency-segment unprotected index 1;
```

Static adjacency SIDs can be configured per area and also based on whether the protection is required or not. Adjacency SIDs should be configured per interface at the [`edit protocols ospf area` *`area`* `interface` *`interface-name`*] hierarchy level.

- Protected—Ensures adjacency SID is eligible to have a backup path and a B-flag is set in an adjacency SID advertisement.
- Unprotected—Ensures no backup path is calculated for a specific adjacency SID and a B-flag is not set in an adjacency SID advertisement.

The following is a sample configuration for IS-IS:

```
user@host# set protocols isis interface ge-0/0/0.1 level 1 ipv4-adjacency-segment unprotected index 1;
user@host# set protocols isis interface ge-0/0/1.1 level 1 ipv4-adjacency-segment protected index 2;
```

The following is a sample configuration for OSPF:

```
user@host# set protocols ospf area 0.0.0.0 interface ge-0/0/0.1 ipv4-adjacency-segment unprotected index 1;
user@host# set protocols ospf area 0.0.0.0 interface ge-0/0/1.1 ipv4-adjacency-segment protected index 2;
```

You can use the same adjacent SID for multiple interfaces by grouping a set of interfaces under an interface group and configuring the adjacency SID for that interface group and traffic can be load balanced among the interfaces under the interface group using weight. This can be configured under the [`edit protocols isis interface-group` *`interface_group_name`*] hierarchy level.

When segment routing is used in LAN subnetworks, each router in the LAN may advertise the adjacency SID of each of its neighbors. To configure adjacency SID for a LAN interface to a specific neighbor, configure the adjacency SIDs under the lan-neighbor configuration at the [`edit protocols isis interface` *`interface_name`* `level` *`level_num`* `lan-neighbor` *`neighbor-sysid`*] hierarchy level for IS-IS and [[`edit protocols ospf area 0.0.0.0 interface` *`interface_name`* `lan-neighbor` *`neighbor-routerid`*]] hierarchy level for OSPF. The following is a sample configuration for IS-IS:

```
user@host# set protocols mpls label-range static-label-range 700000 799999;
user@host# set protocols isis source-packet-routing srgb start-label 800000 index-range 4000;
user@host# set protocols isis interface ge-0/0/0.1 level 1 lan-neighbor 1234.1234.1234 ipv4-adjacency-
segment unprotected label 700001;
```

The following is a sample configuration for OSPF:

```
user@host# set protocols mpls label-range static-label-range 700000 799999;
user@host# set protocols ospf source-packet-routing srgb start-label 800000 index-range 4000;
user@host# set protocols ospf area 0.0.0.0 interface ge-1/0/0.1 lan-neighbor 11.12.1.2 ipv4-adjacency-
segment unprotected label 700001;
```

An adjacency set can be configured by declaring a set of interfaces under an interface group and configuring the adjacency segment for that interface group. The adjacency SID can be picked from the reserved static label pool or ISIS SRGB. Unlike normal interfaces, dynamic adjacency SID is not allocated by default under interface group, in which case the `dynamic` CLI statement is configured. Interfaces configured under an interface group can also be configured separately as independent interfaces as long as the link-group-protection is not configured. The following is a sample configuration:

```
user@host# set protocols mpls label-range static-label-range 700000 799999;
user@host# set protocols isis source-packet-routing srgb start-label 800000 index-range 4000;
user@host# set protocols isis interface-group group1 interface ge-0/0/0.1 weight 1;
user@host# set protocols isis interface-group group1 interface ge-0/0/1.1 weight 2;
user@host# set protocols isis interface-group group1 ipv4-adjacency-segment unprotected label 700001;
```

Use the following CLI hierarchy for configuring adjacency SID for IS-IS:

```
[edit ]
protocols {
    isis {
        interface <interface_name> {
            level <level_num> {
                ipv4-adjacency-segment {
                    protected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                    unprotected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                }
                ipv6-adjacency-segment {
                    protected {
```

```
                    dynamic;
                    label <value>
                    index <index>
                }
                unprotected {
                    dynamic;
                    label <value>
                    index <index>
                }
            }
        }
    }
    interface <interface_name> {
        level <level_num> {
            lan-neighbor <neighbor-sysid>{
                ipv4-adjacency-segment {
                    protected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                    unprotected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                }
                ipv6-adjacency-segment {
                    protected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                    unprotected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                }
            }
        }
    }
    interface-group <interface_group_name> {
```

```
            interface <interface_1> weight <weight>
            ...
            interface <interface_n> weight <weight>
            level <level_num> {
                ipv4-adjacency-segment {
                    protected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                    unprotected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                }
                ipv6-adjacency-segment {
                    protected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                    unprotected {
                        dynamic;
                        label <value>
                        index <index>
                    }
                }
            }
        }
    }
}
```

Use the following CLI hierarchy for configuring adjacency SID for OSPF:

```
[edit ]
protocols {
    ospf {
        area 0.0.0.0 {
        interface <interface_name> {
            ipv4-adjacency-segment {
                protected {
```

```
                    dynamic;
                    label <value>
                    index <index>
                }
                unprotected {
                    dynamic;
                    label <value>
                    index <index>
                }
            }
        }
        interface <interface_name> {
                lan-neighbor <neighbor-routerid>{
                    ipv4-adjacency-segment {
                        protected {
                            dynamic;
                            label <value>
                            index <index>
                        }
                        unprotected {
                            dynamic;
                            label <value>
                            index <index>
                        }
                    }
                }
            }
        }
    }
}
```

Use the following operational CLI commands to verify the configuration:

show isis adjacency detail

The following sample output displays the details of configured and dynamic adjacency SID.

```
user@host> show isis adjacency r1 detail
r1
  Interface: ge-0/0/2.1, Level: 1, State: Up, Expires in 19 secs
  Priority: 64, Up/Down transitions: 1, Last transition: 01:23:38 ago
  Circuit type: 3, Speaks: IP, IPv6, MAC address: 0:5:86:48:49:0
  Topologies: Unicast
```

```
  Restart capable: Yes, Adjacency advertisement: Advertise
  LAN id: r0.03, IP addresses: 11.1.1.2
  IPv6 addresses: fe80::205:8600:148:4900
  Level 1 IPv4   protected Adj-SID: 4138, Flags: BVL
  Level 1 IPv6 unprotected Adj-SID: 4139, Flags: FVL
```

show ospf neighbor detail

The following sample output displays the details of configured and dynamic adjacency SID.

```
user@host> show ospf neighbor detail
Address          Interface              State     ID               Pri  Dead
11.12.1.2        ge-1/0/0.0             Full      12.1.1.1         128    34
  Area 0.0.0.0, opt 0x52, DR 0.0.0.0, BDR 0.0.0.0
  Up 00:06:27, adjacent 00:06:27
  SPRING Adjacency Labels:

     Label       Flags       Adj-Sid-Type

     90010       BVLP        Protected

     1212        VLP         UnProtected
regress@10.49.129.231# run show route label 90010

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

90010              *[L-OSPF/10/5] 00:00:21, metric 0
                    >  to 11.12.1.2 via ge-1/0/0.0, Pop
                       to 11.12.2.2 via ge-1/0/2.0, Swap 16021
                       to 11.12.3.2 via ge-1/0/3.0, Swap 16021
```

show isis database extensive

The following sample output displays the details of LAN/PTP adjacency SID.

```
user@host> show isis database extensive

r0.00-00 Sequence: 0x16, Checksum: 0xf156, Lifetime: 960 secs
  IPV4 Index: 1000, IPV6 Index: 2000
  Node Segment Blocks Advertised:
    Start Index : 0, Size : 4096, Label-Range: [ 16, 4111 ]
```

```
    IS neighbor: r4.00                          Metric:       10
      Two-way fragment: r4.00-00, Two-way first fragment: r4.00-00
    IS neighbor: r0.03                          Metric:       10
      Two-way fragment: r0.03-00, Two-way first fragment: r0.03-00
    IP prefix: 10.10.10.10/32                   Metric:        0 Internal Up
    IP prefix: 11.1.1.0/24                      Metric:       10 Internal Up
    IP prefix: 21.1.1.0/24                      Metric:       10 Internal Up
    V6 prefix: 1001::/64                        Metric:       10 Internal Up
    V6 prefix: 2001::/64                        Metric:       10 Internal Up
    V6 prefix: abcd::10:10:10:10/128            Metric:        0 Internal Up


…
  TLVs:
    Area address: 49.00 (2)
    LSP Buffer Size: 1492
    Speaks: IP
    Speaks: IPV6
    IP router id: 10.10.10.10
    IP address: 10.10.10.10
    Hostname: r0
    IS neighbor: r0.03, Internal, Metric: default 10
    IS neighbor: r4.00, Internal, Metric: default 10
    IS extended neighbor: r0.03, Metric: default 10
      IP address: 11.1.1.1
      Local interface index: 342, Remote interface index: 0
      Current reservable bandwidth:
        Priority 0 : 1000Mbps
        Priority 1 : 1000Mbps
        Priority 2 : 1000Mbps
        Priority 3 : 1000Mbps
        Priority 4 : 1000Mbps
        Priority 5 : 1000Mbps
        Priority 6 : 1000Mbps
        Priority 7 : 1000Mbps
      Maximum reservable bandwidth: 1000Mbps
      Maximum bandwidth: 1000Mbps
         Administrative groups:  0 <none>
      LAN IPV4 Adj-SID: 4138, Weight:0, Neighbor:r1, Flags: BVL
      LAN IPV6 Adj-SID: 4139, Weight:0, Neighbor:r1, Flags: FBVL
    IS extended neighbor: r4.00, Metric: default 10
      IP address: 21.1.1.1
      Neighbor's IP address: 21.1.1.2
      Local interface index: 334, Remote interface index: 335
```

```
      Current reservable bandwidth:
        Priority 0 : 1000Mbps
        Priority 1 : 1000Mbps
        Priority 2 : 1000Mbps
        Priority 3 : 1000Mbps
        Priority 4 : 1000Mbps
        Priority 5 : 1000Mbps
        Priority 6 : 1000Mbps
        Priority 7 : 1000Mbps
      Maximum reservable bandwidth: 1000Mbps
      Maximum bandwidth: 1000Mbps
      Administrative groups:  0 <none>
      P2P IPV4 Adj-SID - Flags: BVL,   Weight:0, Label: 4125
      P2P IPV6 Adj-SID - Flags: FBVL,  Weight:0, Label: 4126
```

show isis interface-group

The following sample output displays the status information about the specified interface group.

```
user@host> show isis interface-group
Interface-group: r1r2ig
  ge-0/0/1.1, 1000Mbps, Up, Non-Degraded, Weight: 1
  ge-0/0/1.3, 1000Mbps, Up, Non-Degraded, Weight: 1
  ge-0/0/1.5, 1000Mbps, Up, Non-Degraded, Weight: 1
  Total Nominal Bandwidth: 3Gbps, Total Actual Bandwidth: 3Gbps
  Level 1 IPv4   protected Adj-SID: Label 4138
  Level 1 IPv6 unprotected Adj-SID: Label 4139
```

Topology-Independent Loop-Free Alternate with Segment Routing for IS-IS and OSPF

IN THIS SECTION

Segment routing enables a router to send a packet along a specific path in the network by imposing a label stack that describes the path. The forwarding actions described by a segment routing label stack do not need to be established on a per-path basis. Therefore, an ingress router can instantiate an arbitrary path using a segment routing label stack and use it immediately without any signaling.

In segment routing, each node advertises mappings between incoming labels and forwarding actions. A specific forwarding action is referred to as a segment and the label that identifies that segment is referred to as a segment identifier (SID). The backup paths created by TI-LFA use the following types of segments:

- Node segment—A node segment forwards packets along the shortest path or paths to a destination node. The label representing the node segment (the node SID) is swapped until the destination node is reached.
- Adjacency segment—An adjacency segment forwards packets across a specific interface on the node that advertised the adjacency segment. The label representing an adjacency segment (the adjacency SID) is popped by the node that advertised it.

A router can send a packet along a specific path by creating a label stack that uses a combination of node SIDs and adjacency SIDs. Typically, node SIDs are used to represent parts of the path that correspond to the shortest path between two nodes. An adjacency SID is used wherever a node SID cannot be used to accurately represent the desired path.

Loop-free alternate (LFA) and remote LFA (RLFA) have been used to provide fast-reroute protection for several years. With LFA, a point of local repair (PLR) determines whether or not a packet sent to one of its direct neighbors reaches its destination without looping back through the PLR. In a typical network topology, approximately 40 to 60 percent of the destinations can be protected by LFA. Remote LFA expands on the concept of LFA by allowing the PLR to impose a single label to tunnel the packet to a repair tunnel endpoint from which the packet can reach its destination without looping back through the PLR. Using remote LFA, more destinations can be protected by the PLR compared to LFA. However, depending on the network topology, the percentage of destinations protected by remote LFA is usually less than 100 percent.

Topology-independent LFA (TI-LFA) extends the concept of LFA and remote LFA by allowing the PLR to use deeper label stacks to construct backup paths. In addition, the TI-LFA imposes the constraint that the backup path used by the PLR be the same path that a packet takes once the interior gateway

protocol (IGP) has converged for a given failure scenario. This path is referred to as the post-convergence path.

Using the post-convergence path as the backup path has some desirable characteristics. For some topologies, a network operator only needs to make sure that the network has enough capacity to carry the traffic along the post-convergence path after a failure. In these cases, a network operator does not need to allocate additional capacity to deal with the traffic pattern immediately after the failure while the backup path is active, because the backup path follows the post-convergence path.

Benefits of TI-LFA

- IGP automatically computes the backup path and the backup path follows the post failure path. You must plan capacity for post failure path and not allocate separate capacity for backup paths.
- Provides redundancy and protects against link failure.
- Easy to configure and utilize the post convergence path for transmission of packets.

Types of TI-LFA Protection

TI-LFA provides protection against link failure, node failure, fate-sharing failures, and shared risk link group failures. In link failure mode, the destination is protected if the link fails. In node protection mode, the destination is protected if the neighbor connected to the primary link fails. To determine the node-protecting post-convergence path, the cost of all the links leaving the neighbor is assumed to increase by a configurable amount.

With fate-sharing protection, a list of fate-sharing groups are configured on each PLR with the links in each fate-sharing group identified by their respective IP addresses. The PLR associates a cost with each fate-sharing group. The fate-sharing-aware post-convergence path is computed by assuming that the cost of each link in the same fate-sharing group as the failed link has increased the cost associated with that group.

Configure Shared Risk Link Group (SRLG) protection in TI-LFA networks for segment routing to choose a fast reroute path that does not include SRLG links in the topology-independent loop-free alternate (TI-LFA) backup paths. SRLGs share a common fibre and they also share the risks of a broken link. When one link in an SRLG fails, other links in the group might also fail. Therefore, you need to avoid links that share the same risk as the protected link in the backup path. Configuring SRLG protection prevents TI-LFA from selecting backup paths that include a shared risk link. If you have configured SRLG protection then IS-IS computes the fast reroute path that is aligned with the post convergence path and excludes the links that belong to the SRLG of the protected link. All local and remote links that are from the same SRLG as the protected link are excluded from the TI-LFA back up path. The point of local repair (PLR) sets up the label stack for the fast reroute path with a different outgoing interface. Currently you cannot enable SRLG protection in IPv6 networks and in networks with multitopology.

In order to construct a backup path that follows the post-convergence path, TI-LFA uses several labels in the label stack that define the backup path. If the number of labels required to construct a particular post-convergence backup path exceeds a certain amount, it is useful in some circumstances to not install that backup path. You can configure the maximum number of labels that a backup path can have in order to be installed. The default value is 3, with a range of 2 through 5.

It is often the case that the post-convergence path for a given failure is actually a set of equal-cost paths. TI-LFA attempts to construct the backup paths to a given destination using multiple equal-cost paths in the post-failure topology. Depending on the topology, TI-LFA might need to use different label stacks to accurately construct those equal-cost backup paths. By default, TI-LFA only installs one backup path for a given destination. However, you can configure the value in the range from 1 through 8.

TI-LFA in IPv6 Networks

Configure TI-LFA with segment routing in an IPv6-only network to provide fast reroute (FRR) backup paths corresponding to the post-convergence path for a given failure. However, you cannot configure fate-sharing protection for IPv6-only networks. To compute backup paths in IPv6-only networks, the IS-IS protocol must advertise the following TLV types:

- TLV 233 - IPv6 Global Interface Address
- Subtlv 12 and 13 of TLV 22

You can configure multiple global IPv6 addresses on the interface. If you configure the `use-source-packet-routing` statement, then all addresses are assigned TI-LFA backup paths.

Configure a point of local repair (PLR) to create a topology independent loop-free alternate backup path for prefix-SIDs derived from Segment Routing Mapping Server advertisements in an IS-IS network. In a network configured with segment routing, IS-IS uses the Segment Routing Mapping Server advertisements to derive prefix-SIDs. Segment Routing Mapping Server advertisements for IPv6 are currently not supported. To attach flags to Segment Routing Mapping Server advertisements, include the `attached`, `domain-wide-flooding`, and `no-node-segment` statements at the `[edit routing-options source-packet-routing mapping-server-entry` *`mapping-server-name`*`]` hierarchy level.

TI-LFA Limitations

The backup path for prefix-SIDs from Segment Routing Mapping Server advertisements are not created in the following scenarios:

- If some hops are present in a non-SR domain.
- If the segment routing node is advertising a prefix and a prefix-SID index directly, then Junos OS uses the prefix-SID index and disregards the mapping server advertisement for that prefix.

- If a backup path requires an adjacency-SID from the LDP domain then the backup path cannot be installed.
- If the PLR is unable to determine the label mapping using LDP.

> **NOTE**: Currently you cannot configure remote LFA and TI-LFA on a SR-LDP stitching node in the same instance. Therefore, you cannot configure both `post-convergence-lfa` and `link-protection` on the same device.

Advertisement Flags for TI-LFA

Set the following mapping server advertisement flags to indicate the origin of the advertised prefix:

Flag	TLV Name	Flag Values	Length	Description
A	Label Binding TLV	0, 1 default value is 0	1	Attached Flag–Include the `attached` configuration statement to set this flag to 1 to indicate that the prefixes and SIDs advertised in the SID or Label Binding TLV are directly connected to their originators.
S	Label Binding TLV	0, 1 default value is 0	1	Include the `domain-wide-flooding` configuration statement to set this flag to 1 to indicate that the SID or Label Binding TLV is flooded across the entire routing domain.
D	Label Binding TLV	0, 1 default value is 0	1	Set by a border node when readvertising a SID or Label Binding TLV to indicate that the SID or Label Binding TLV is leaked from level 2 to level 1.
N	Prefix-SID sub TLV	0, 1 default value is 1	1	Include the `no-node-segment` configuration statement to set this flag to 0 to indicate that the prefix has originated from a single node.

RELATED DOCUMENTATION

Configuring Topology-Independent Loop-Free Alternate with Segment Routing for IS-IS

Example: Configuring Topology Independent Loop-Free Alternate with Segment Routing for IS-IS

post-convergence-lfa

use-post-convergence-lfa

use-for-post-convergence-lfa

node-protection

Configuring Topology-Independent Loop-Free Alternate with Segment Routing for IS-IS

Loop-free alternate (LFA) and remote LFA have been used to provide fast-reroute protection for several years. With LFA, a point of local repair (PLR) determines whether or not a packet sent to one of its direct neighbors will reach its destination without looping back through the PLR. In a typical network topology, perhaps 40-60 percent of destinations can be protected by LFA. Remote LFA expands on the concept of LFA by allowing the PLR to impose a single label to tunnel the packet to a repair tunnel endpoint from which the packet can reach its destination without looping back through the PLR. Using remote LFA, more destinations can be protected by the PLR compared to LFA. However, depending on the network topology, the percentage of destinations protected by remote LFA usually less than 100 percent.

Topology-independent loop-free alternate (TI-LFA) extends the concept of LFA and remote LFA by allowing the PLR to use deeper label stacks to construct backup paths. In addition, TI-LFA imposes the constraint that the backup path used by the PLR be the same path that a packet takes once the IGP converges for a given failure scenario. This path is referred to as the post-convergence path.

Using the post-convergence path as the backup path has some desirable characteristics. For some topologies, a network operator only needs to make sure that the network has enough capacity to carry the traffic along the post-convergence path after a failure. In these cases, a network operator does not need to allocate additional capacity to deal with the traffic pattern immediately after the failure while the backup path is active, because the backup path follows the post-convergence path.

Before you configure TI-LFA for IS-IS, be sure you configure SPRING or segment routing.

To configure TI-LFA using SPRING for IS-IS, you must do the following:

1. Enable TI-LFA for IS-IS protocol.

```
[edit protocols isis backup-spf-options]
user@R1# set use-post-convergence-lfa
```

2. (Optional) Configure backup shortest path first (SPF) attributes such as maximum equal-cost multipath (ECMP) backup paths and maximum labels for TI-LFA for the IS-IS protocol.

```
[edit protocols isis backup-spf-options use-post-convergence-lfa]
user@R1# set maximum-backup-paths maximum-backup-paths
user@R1# set maximum-labels maximum-labels
```

3. Configure the computation and installation of a backup path that follows the post-convergence path on the given interface and level for the IS-IS protocol.

```
[edit protocols isis interface interface-name level level
user@R1# set post-convergence-lfa
```

4. (Optional) Enable fate-sharing protection for a given interface and level. Specify the fate-sharing group to use as a constraint for the post-convergence path.

> **NOTE**: You do not have to configure the `use-for-post-convergence-lfa` statement and the `fate-sharing-protection` statement for basic link protection for the backup path.

```
[edit routing-options fate-sharing group group-name]
user@R1# set use-for-post-convergence-lfa
```

```
[edit protocols isis interface interface-name level level post-convergence-lfa]
user@R1# set fate-sharing-protection
```

5. (Optional) Enable node protection for a given interface and level.

```
[edit protocols isis interface interface-name level level post-convergence-lfa]
user@R1# set node-protection
```

RELATED DOCUMENTATION

Understanding Topology-Independent Loop-Free Alternate with Segment Routing for IS-IS

Example: Configuring Topology Independent Loop-Free Alternate with Segment Routing for IS-IS

post-convergence-lfa

use-post-convergence-lfa

use-for-post-convergence-lfa

node-protection

Example: Configuring Topology Independent Loop-Free Alternate with Segment Routing for IS-IS

IN THIS SECTION

This example shows topology-independent loop-free alternate (TI-LFA) with segment routing for the IS-IS protocol to provide MPLS fast reroute (FRR) backup paths corresponding to the post-convergence path for a given failure by using deeper label stacks to construct backup paths. TI-LFA provides protection against link failure, node failure, and fate-sharing failures. In link failure mode, the destination is protected if the link fails. In node protection mode, the destination is protected if the neighbor connected to the primary link fails. To determine the node-protecting post-convergence path, the cost of all the links leaving the neighbor is assumed to increase by a configurable amount. With fate-sharing protection, a list of fate-sharing groups are configured on each PLR with the links in each fate-sharing group identified by their respective IP addresses.

NOTE: Our content testing team has validated and updated this example.

Requirements

This example uses the following hardware and software components:

- Nine MX Series routers
- Junos OS Release 17.4 or later running on all devices
  - Updated and revalidated using vMX on Junos OS Release 21.1R1.

Before you configure TI-LFA routes using SPRING for IS-IS, be sure you configure SPRING or segment routing.

NOTE: Are you interested in getting hands-on experience on this feature?
Visit Juniper vLabs to reserve your pre-configured vLab Sandbox: Segment Routing - Basic and try it out for free!

Overview

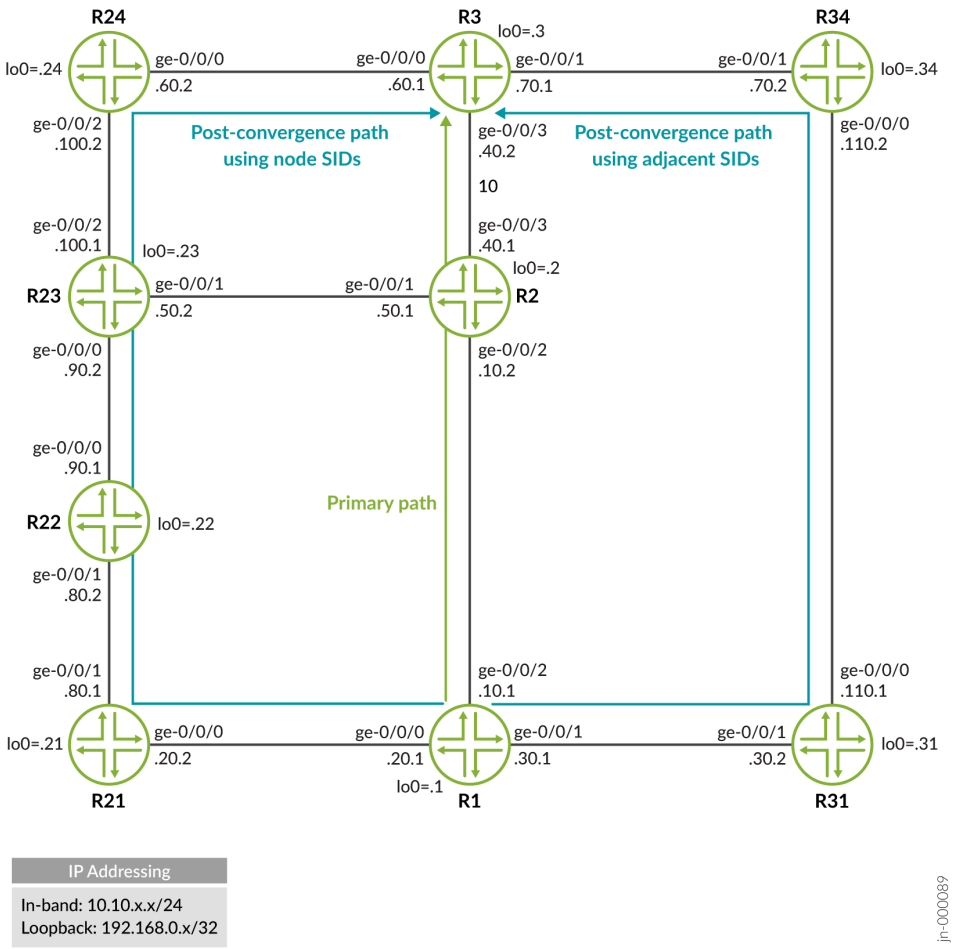
IN THIS SECTION

Junos OS allows you to enable TI-LFA for IS-IS by configuring the `use-post-convergence-lfa` statement at the `[edit protocols isis backup-spf-options]` hierarchy level. You can enable the creation of post-convergence backup paths for a given interface by configuring the post-convergence-lfa statement at the `[edit protocols isis interface interface-name level level]` hierarchy level.

TI-LFA provides protection against link failure, node failure, and failures of fate-sharing groups. You can enable link-protection mode using the `post-convergence-lfa` statement. You can enable `node-protection` mode, or `fate-sharing-protection` mode, or both modes, for a given interface at the `[edit protocols isis interface interface-name level level post-convergence-lfa]` hierarchy level. To ensure that the fate-sharing protection is enabled for a given fate-sharing group, you need to configure the `use-for-post-convergence-lfa` statement at the `[edit routing-options fate-sharing group group-name]` hierarchy level.

NOTE: TI-LFA supports protection of routes for both IPv4 and IPv6 prefixes. This example demonstrates protection of routes for IPv4 prefixes.

Topology



Configuration

IN THIS SECTION

- Verification | 98

CLI Quick Configuration

To quickly configure link-protection in this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration,

copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

R1

```
set interfaces ge-0/0/0 unit 0 description r1-to-r21
set interfaces ge-0/0/0 unit 0 family inet address 10.10.20.1/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description r1-to-r31
set interfaces ge-0/0/1 unit 0 family inet address 10.10.30.1/24
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 unit 0 description r1-to-r2
set interfaces ge-0/0/2 unit 0 family inet address 10.10.10.1/24
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 198.168.0.1/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0001.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 disable
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 level 1 post-convergence-lfa
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv4-index 1001
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set routing-options router-id 198.168.0.1
```

R2

```
set interfaces ge-0/0/1 unit 0 description r2-to-r23
set interfaces ge-0/0/1 unit 0 family inet address 10.10.50.1/24
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 unit 0 description r2-to-r1
set interfaces ge-0/0/2 unit 0 family inet address 10.10.10.2/24
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 unit 0 description r2-to-r3
set interfaces ge-0/0/3 unit 0 family inet address 10.10.40.1/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0002.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/1.0 level 2 disable
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 2 disable
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv4-index 1002
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set protocols mpls interface ge-0/0/3.0
set routing-options router-id 192.168.0.2
```

R3

```
set interfaces ge-0/0/0 unit 0 description r3-to-r24
set interfaces ge-0/0/0 unit 0 family inet address 10.10.60.1/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description r3-to-r34
set interfaces ge-0/0/1 unit 0 family inet address 10.10.70.1/24
```

```
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/3 unit 0 description r3-to-r2
set interfaces ge-0/0/3 unit 0 family inet address 10.10.40.2/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0003.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 disable
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 2 disable
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv4-index 1003
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/3.0
set routing-options router-id 192.168.0.3
```

R21

```
set interfaces ge-0/0/0 unit 0 description r21-to-r1
set interfaces ge-0/0/0 unit 0 family inet address 10.10.20.2/24
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 unit 0 description r21-to-r22
set interfaces ge-0/0/1 unit 0 family inet address 10.10.80.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.21/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0021.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 disable
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface fxp0.0 disable
```

```
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv4-index 1021
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set routing-options router-id 192.168.0.21
```

R22

```
set interfaces ge-0/0/0 unit 0 description r22-to-r23
set interfaces ge-0/0/0 unit 0 family inet address 10.10.90.1/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description r22-to-r21
set interfaces ge-0/0/1 unit 0 family inet address 10.10.80.2/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.22/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0022.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 disable
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv4-index 1022
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set routing-options router-id 192.168.0.22
```

R23

```
set interfaces ge-0/0/0 unit 0 description r23-to-r22
set interfaces ge-0/0/0 unit 0 family inet address 10.10.90.2/24
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 unit 0 description r23-to-r2
set interfaces ge-0/0/1 unit 0 family inet address 10.10.50.2/24
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 unit 0 description r23-to-r24
set interfaces ge-0/0/2 unit 0 family inet address 10.10.100.1/24
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 192.168.0.23/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0023.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 disable
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv4-index 1023
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set routing-options router-id 192.168.0.23
```

R24

```
set interfaces ge-0/0/0 unit 0 description r24-to-r3
set interfaces ge-0/0/0 unit 0 family inet address 10.10.60.2/24
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 unit 0 description r24-to-r23
set interfaces ge-0/0/2 unit 0 family inet address 10.10.100.2/24
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 192.168.0.24/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0024.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
```

```
set protocols isis source-packet-routing node-segment ipv4-index 1024
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/2.0
set routing-options router-id 192.168.0.24
```

R31

```
set interfaces ge-0/0/0 unit 0 description r31-to-r34
set interfaces ge-0/0/0 unit 0 family inet address 10.10.110.1/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description r31-to-r1
set interfaces ge-0/0/1 unit 0 family inet address 10.10.30.2/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.162.0.31/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0031.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/0.0 level 1 metric 500
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 1 metric 10
set protocols isis interface ge-0/0/1.0 level 2 disable
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv4-index 1031
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/0.0
set routing-options router-id 198.162.0.31
```

R34

```
set interfaces ge-0/0/0 unit 0 description r34-to-r31
set interfaces ge-0/0/0 unit 0 family inet address 10.10.110.2/24
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 unit 0 description r34-to-r3
set interfaces ge-0/0/1 unit 0 family inet address 10.10.70.2/24
```

```
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.34/32
set interfaces lo0 unit 0 family iso address 49.0000.2222.0034.00
set interfaces lo0 unit 0 family mpls
set protocols isis interface ge-0/0/0.0 level 1 metric 500
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 1 metric 10
set protocols isis interface ge-0/0/1.0 level 2 disable
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv4-index 1034
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set routing-options router-id 192.168.0.34
```

Configuring R1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set ge-0/0/0 unit 0 description r1-to-r21
user@R1# set ge-0/0/0 unit 0 family inet address 10.10.20.1/24
user@R1# set ge-0/0/0 unit 0 family iso
user@R1# set ge-0/0/0 unit 0 family mpls

user@R1# set ge-0/0/1 unit 0 description r1-to-r31
user@R1# set ge-0/0/1 unit 0 family inet address 10.10.30.1/24
user@R1# set ge-0/0/1 unit 0 family iso
user@R1# set ge-0/0/1 unit 0 family mpls

user@R1# set ge-0/0/2 unit 0 description r1-to-r2
```

```
user@R1# set ge-0/0/2 unit 0 family inet address 10.10.10.1/24
user@R1# set ge-0/0/2 unit 0 family iso
user@R1# set ge-0/0/2 unit 0 family mpls

user@R1# set lo0 unit 0 family inet address 198.168.0.1/32
user@R1# set lo0 unit 0 family iso address 49.0000.2222.0001.00
user@R1# set lo0 unit 0 family mpls
```

2. Configure the router ID.

```
[edit routing-options]
user@R1# set router-id 198.168.0.1
```

3. Configure MPLS.

```
[edit protocols]
user@R1# set mpls interface ge-0/0/0.0
user@R1# set mpls interface ge-0/0/1.0
user@R1# set mpls interface ge-0/0/2.0
```

4. Configure IS-IS.

```
[edit protocols]
user@R1# set isis interface ge-0/0/0.0 level 2 disable
user@R1# set isis interface ge-0/0/0.0 point-to-point

user@R1# set isis interface ge-0/0/1.0 level 2 disable
user@R1# set isis interface ge-0/0/1.0 point-to-point

user@R1# set isis interface ge-0/0/2.0 level 2 disable
user@R1# set isis interface ge-0/0/2.0 point-to-point

user@R1# set isis interface lo0.0 passive

user@R1# set isis interface fxp0.0 disable
```

5. Configure to install backup route along the link-protecting post-convergence path on interface ge-0/0/2.

```
[edit protocols]
user@R1# set isis interface ge-0/0/2.0 level 1 post-convergence-lfa
```

6. Configure the maximum number of labels for segment routing routed paths for protection of backup shortest-path-first attributes.

```
[edit protocols]
user@R1# set isis backup-spf-options use-post-convergence-lfa maximum-labels 8
```

7. Configure IPv4 index and index range for node segments in segment routing for the IS-IS protocol.

```
[edit protocols]
user@R1# set isis source-packet-routing node-segment ipv4-index 1001
```

8. (Optional) Enable node-protection on interface ge-0/0/2.

```
[edit protocols]
user@R1# set isis interface ge-0/0/2 level 2 post-convergence-lfa node-protection cost 2000
```

9. (Optional) Configure the fate-sharing group cost.

```
[edit routing-options]
user@R1# set fate-sharing group fs-group-1 cost 3000
```

10. (Optional) Configure the fate-sharing group to indicate that link from Device R1 to Device R2 and the link from Device R21 to Device R22 share fate and allow it to be used for post-convergence-lfa.

```
[edit routing-options]
user@R1# set fate-sharing group fs-group-1 from 10.10.10.1 to 10.10.10.2
user@R1# set fate-sharing group fs-group-1 from 10.10.80.1 to 10.10.80.2
user@R1# set fate-sharing group fs-group-1 use-for-post-convergence-lfa
```

11. (Optional) Enable fate-sharing protection for ge-0/0/2 on Device R1.

```
[edit protocols]
user@R1# set isis interface ge-0/0/2 level 2 post-convergence-lfa fate-sharing-protection
```

12. Configure a per packet load-balance policy for TI-LFA to work and ensure faster convergence.

```
[edit]
user@R1# set policy-options policy-statement pplb then load-balance per-packet
```

13. Apply the policy to export the routes into the forwarding table.

```
[edit]
user@R1# set routing-options forwarding-table export pplb
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols` and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
ge-0/0/0 {
    unit 0 {
        description r1-to-r21;
        family inet {
            address 10.10.20.1/24;
        }
        family iso;
        family mpls;
    }
}
ge-0/0/1 {
    unit 0 {
        description r1-to-r31;
        family inet {
            address 10.10.30.1/24;
        }
        family iso;
```

```
        family mpls;
    }
}
ge-0/0/2 {
    unit 0 {
        description r1-to-r2;
        family inet {
            address 10.10.10.1/24;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 198.168.0.1/32;
        }
        family iso {
            address 49.0000.2222.0001.00;
        }
        family mpls;
    }
}
```

```
user@R1# show routing-options
router-id 198.168.0.1;
forwarding-table {
    export pplb;
}
```

```
user@R1# show policy-options
policy-statement pplb {
    then {
        load-balance per-packet;
```

```
    }
}
```

```
user@R1# show protocols
isis {
    interface ge-0/0/0.0 {
        level 2 disable;
        point-to-point;
    }
    interface ge-0/0/1.0 {
        level 2 disable;
        point-to-point;
    }
    interface ge-0/0/2.0 {
        level 2 disable;
        level 1 {
            post-convergence-lfa;
        }
        point-to-point;
    }
    interface fxp0.0 {
        disable;
    }
    interface lo0.0 {
        passive;
    }
    source-packet-routing {
        node-segment ipv4-index 1001;
    }
    backup-spf-options {
        use-post-convergence-lfa maximum-labels 8;
    }
}
mpls {
    interface ge-0/0/0.0;
    interface ge-0/0/1.0;
    interface ge-0/0/2.0;
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verify the TI-LFA routes using node SIDs

Purpose

Verify the link-protecting backup path for primary next hops on interface ge-0/0/2 for Device R1 and verify if the backup path to reach 192.168.0.3/32 has been created and has the correct label stack.

Action

From operational mode, run the `show route 192.168.0.3` command to display the routing table information.

```
user@R1> show route 192.168.0.3

inet.0: 38 destinations, 38 routes (38 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.3/32     *[IS-IS/15] 09:52:56, metric 20
                    >  to 10.10.10.2 via ge-0/0/2.0

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

192.168.0.3/32     *[L-ISIS/14] 05:45:40, metric 20
                    >  to 10.10.10.2 via ge-0/0/2.0, Push 801003
                       to 10.10.20.2 via ge-0/0/0.0, Push 801003, Push 801024(top)
```

Meaning

The primary path to reach 198.168.0.3/32 (corresponding to Device R3) is through the interface ge-0/0/2 with a label of 801003, corresponding to the node-SID of Device R3. If the interface ge-0/0/2 fails, the backup path using the interface ge-0/0/0 using the label stack [801024, 801003] becomes active. The link-protecting post-convergence path is R1-R21-R22-R23-R24-R3. The top label on the label stack is 801024 and corresponds to the node SID to reach R24. The 801003 label corresponds to the node SID on R23 to reach R3 on the shortest path R23-R2-R3.

Verify adjacency SIDs

Verify adjacency SIDs of devices that have IS-IS adjacencies with Device R1.

NOTE: The SID values can vary in your configuration setup.

Action

From operational mode, run the `show isis adjacency detail` command to display the adjacency information on Device R1.

```
user@R1> show isis adjacency detail
R21
  Interface: ge-0/0/0.0, Level: 1, State: Up, Expires in 19 secs
  Priority: 0, Up/Down transitions: 3, Last transition: 07:06:07 ago
  Circuit type: 1, Speaks: IP, IPv6
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  IP addresses: 10.10.20.2
  Level 1 IPv4 Adj-SID: 299840

R31
  Interface: ge-0/0/1.0, Level: 1, State: Up, Expires in 22 secs
  Priority: 0, Up/Down transitions: 3, Last transition: 07:06:07 ago
  Circuit type: 1, Speaks: IP, IPv6
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  IP addresses: 10.10.30.2
  Level 1 IPv4 Adj-SID: 299808

R2
  Interface: ge-0/0/2.0, Level: 1, State: Up, Expires in 24 secs
```

```
  Priority: 0, Up/Down transitions: 3, Last transition: 07:06:07 ago
  Circuit type: 1, Speaks: IP, IPv6
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  IP addresses: 10.10.10.2
  Level 1 IPv4 Adj-SID: 299776
```

Meaning

Adjacency SIDs are assigned to each adjacency of Device R1 in the segment routing domain:

- Device R21 - 299840
- Device R31 - 299808
- Device R2 - 299776

The adjacency SIDs have local significance and can be used to steer traffic along specific outgoing interfaces. When you do not configure adjacency SIDs, they are dynamically assigned with a value outside of the default (or configured) SRGB range.

Verify the TI-LFA routes using adjacency SIDs

Purpose

Increase the cost of the post-convergence path from R1 to R3 and verify the TI-LFA routes using adjacency SIDs to avoid the primary path to reach the destination, Device R3.

Action

From the configuration mode, increase the cost of the interface connecting Device R22 and R23, ge-0/0/0.

```
[edit protocols]
user@R22# set protocols isis interface ge-0/0/0.0 level 1 metric 1000
user@R22# commit
```

From operational mode, again run the `show route 192.168.0.3` command.

```
user@R1> show route 192.168.0.3
```

```
inet.0: 38 destinations, 38 routes (38 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.3/32     *[IS-IS/15] 10:44:56, metric 20
                    >  to 10.10.10.2 via ge-0/0/2.0

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

192.168.0.3/32     *[L-ISIS/14] 00:00:31, metric 20
                    >  to 10.10.10.2 via ge-0/0/2.0, Push 801003
                       to 10.10.30.2 via ge-0/0/1.0, Push 801003, Push 299808(top)
```

Meaning

The TI-LFA backup paths are now using the adjacency SID (in this case, 299808) instead of the node SID (801003) to reach Device R3. This is because node SIDs always use the shortest path between two nodes, and when the R22-R23 link cost went up, the shortest path to R1 overlaps with the primary path. Because TI-LFA cannot take a primary path to reach the destination, adjacency SIDs are used to take R31-R34 as the new post-convergence path to reach Device R3.

RELATED DOCUMENTATION

- *Understanding Topology-Independent Loop-Free Alternate with Segment Routing for IS-IS*
- *Configuring Topology-Independent Loop-Free Alternate with Segment Routing for IS-IS*
- *post-convergence-lfa*
- *use-post-convergence-lfa*
- *use-for-post-convergence-lfa*
- *node-protection*
- vLab Sandbox: Segment Routing - Basic

Topology-Independent Loop-Free Alternate with Segment Routing for OSPF

IN THIS SECTION

Topology-Independent Loop-Free Alternate with Segment Routing for OSPF Overview

IN THIS SECTION

This section describes the TI-LFA feature for OSPF.

When used with OSPF, TI-LFA provides protection against link failure, node failure., fate-sharing failures, and shared risk link group failures. In link failure mode, the destination is protected if the link fails. In node protection mode, the destination is protected if the neighbor connected to the primary link fails. To determine the node-protecting post-convergence path, the cost of all the links leaving the neighbor is assumed to increase by a configurable amount.

Configure fate-sharing protection in TI-LFA networks for segment routing to choose a fast reroute path that does not include fate-sharing groups in the topology-independent loop-free alternate (TI-LFA) backup paths to avoid fate-sharing failures. With fate-sharing protection, a list of fate-sharing groups are configured on each PLR with the links in each fate-sharing group identified by their respective IP addresses. The PLR associates a cost with each fate-sharing group. The fate-sharing-aware post-convergence path is computed by assuming that the cost of each link in the same fate-sharing group as the failed link has increased the cost associated with that group.

Configure Shared Risk Link Group (SRLG) protection in TI-LFA networks for segment routing to choose a fast reroute path that does not include SRLG links in the topology-independent loop-free alternate (TI-LFA) backup paths. SRLGs share a common fibre and they also share the risks of a broken link. When one link in an SRLG fails, other links in the group might also fail. Therefore, you need to avoid links that share the same risk as the protected link in the backup path. Configuring SRLG protection prevents TI-

LFA from selecting backup paths that include a shared risk link. If you have configured SRLG protection then OSPFv2 computes the fast reroute path that is aligned with the post convergence path and excludes the links that belong to the SRLG of the protected link. All local and remote links that are from the same SRLG as the protected link are excluded from the TI-LFA back up path. The point of local repair (PLR) sets up the label stack for the fast reroute path with a different outgoing interface. Currently you cannot enable SRLG protection in IPv6 networks and in networks with multitopology.

In order to construct a backup path that follows the post-convergence path, TI-LFA can use several labels in the label stack that define the backup path. If the number of labels required to construct a particular post-convergence backup path exceeds a certain amount, it is useful in some circumstances to not install that backup path. You can configure the maximum number of labels that a backup path can have in order to be installed. The default value is 3, with a range of 2 through 5.

It is often the case that the post-convergence path for a given failure is actually a set of equal-cost paths. TI-LFA attempts to construct the backup paths to a given destination using multiple equal-cost paths in the post-failure topology. Depending on the topology, TI-LFA might need to use different label stacks to accurately construct those equal-cost backup paths. By default, TI-LFA only installs one backup path for a given destination. However, you can configure the value in the range from 1 through 8.

Benefits of Using Topology-Independent Loop-Free Alternate with Segment Routing

- Loop-free alternate (LFA) and remote LFA (RLFA) have been used to provide fast-reroute protection for several years. With LFA, a point of local repair (PLR) determines whether or not a packet sent to one of its direct neighbors reaches its destination without looping back through the PLR. In a typical network topology, approximately 40 to 60 percent of the destinations can be protected by LFA. Remote LFA expands on the concept of LFA by allowing the PLR to impose a single label to tunnel the packet to a repair tunnel endpoint from which the packet can reach its destination without looping back through the PLR. Using remote LFA, more destinations can be protected by the PLR compared to LFA. However, depending on the network topology, the percentage of destinations protected by remote LFA is usually less than 100 percent.

- Topology-independent LFA (TI-LFA) extends the concept of LFA and remote LFA by allowing the PLR to use deeper label stacks to construct backup paths. In addition, TI-LFA imposes the constraint that the backup path used by the PLR be the same path that a packet takes once the interior gateway protocol (IGP) has converged for a given failure scenario. This path is referred to as the post-convergence path.

- Using the post-convergence path as the backup path has some desirable characteristics. For some topologies, a network operator only needs to make sure that the network has enough capacity to carry the traffic along the post-convergence path after a failure. In these cases, a network operator does not need to allocate additional capacity to deal with the traffic pattern immediately after the failure while the backup path is active, because the backup path follows the post-convergence path.

- When used with OSPF, TI-LFA provides protection against link failure and node failure.

Configuring Topology-Independent Loop-Free Alternate with Segment Routing for OSPF

Before you configure TI-LFA for OSPF, be sure you configure SPRING or segment routing.

Junos supports creation of OSPF topology-independent TI-LFA backup paths where the prefix SID is learned from a segment routing mapping server advertisement when the PLR and mapping server are both in the same OSPF area.

To configure TI-LFA using SPRING for OSPF, you must do the following:

1. Enable TI-LFA for OSPF protocol.

```
[edit protocols ospf backup-spf-options]
user@R1# set use-post-convergence-lfa
```

2. (Optional) Configure backup shortest path first (SPF) attributes such as maximum equal-cost multipath (ECMP) backup paths and maximum labels for TI-LFA for the OSPF protocol.

```
[edit protocols ospf backup-spf-options use-post-convergence-lfa]
user@R1# set maximum-backup-paths maximum-backup-paths
user@R1# set maximum-labels maximum-labels
```

3. Configure the computation and installation of a backup path that follows the post-convergence path on the given area and interfacefor the OSPF protocol.

```
[edit protocols ospf area area-id interface interface-name
user@R1# set post-convergence-lfa
```

4. (Optional) Enable node protection for a given area and interface.

```
[edit protocols ospf area area-id interfaceinterface-name post-convergence-lfa]
user@R1# set node-protection
```

5. (Optional) Enable fate-sharing protection for a given area and interface.

```
[edit protocols ospf  area area-id interfaceinterface-name  post-convergence-lfa]
user@R1# set fate-sharing-protection
```

6. (Optional) Enable SRLG protection for a given area and interface.

```
[edit protocols ospf  area area-id interfaceinterface-name  post-convergence-lfa]
user@R1# set srlg-protection
```

RELATED DOCUMENTATION

source-packet-routing

use-post-convergence-lfa

post-convergence-lfa

IGP Microloop Avoidance

SUMMARY

Microloops can consume the available bandwidth of the links, which impacts the efficient transmission of useful packets. Microloop avoidance can prevent forwarding of looping packets.

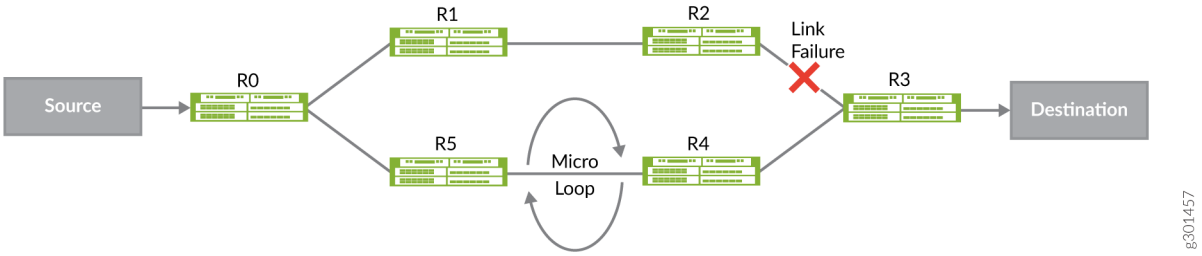
IN THIS SECTION

Benefits of Avoiding Microloops

- Micro loop-free path avoids delays and traffic loss
- Microloop avoidance can prevent forwarding of looping packets and avoid wasteful bandwidth consumption
- Microloop avoidance path is computed only for the impacted links in case of multiple link failures. If the second link failure does not impact the computed microloop avoidance path, IGP continues to use the same microloop avoidance path.

Junos OS enables a device to defer IS-IS route download when an IS-IS link fails in order to avoid micro loops. When local links go down, the IS-IS protocol floods an entire area with the database. If the node connected to the local interface that has failed converges faster than the neighboring node, then the

connected node redirects traffic to the converged path. This redirection can result in micro looping of traffic until the neighboring node converges. When the primary path of a protected node fails, the connected node does not need to converge quickly if the configured backup path is not impacted. In this case, traffic flow towards a converged path is deferred until the configured delay time. This time delay helps in avoiding microloops because all routers do not arrive at the post-convergence forwarding states simultaneously.



In the "figure" on page 105, the primary path from Source to Destination is S→R0→R1→R2→R3→D. When the link between R2 and R3 fails, traffic sent from S to D, is subject to transient forwarding loops while routers update their forwarding state for destination D.

- If R0 updates its forwarding state before R5, packets will loop between R0 and R5
- If both R0 and R5, have updated their forwarding states, and R4 has not, packets will loop between R4 and R5.
- R0 detects the link failure between R2 and R3, and temporarily steers traffic destined to Destination over SR path [NodeSID(R4), AdjSID(R4->R3), D].
- When the configured timeout elapses, R0 just uses the node-SID to D to reach the destination.

Microloop Avoidance in SR-MPLS Networks

Enable post-convergence path calculation on a device to avoid microloops between network devices. Microloops form when a network change such as a link or metric change occurs in a segment routing MPLS network. A network change might trigger a loop between upstream and downstream routers for a brief time period because the routers do not update their forwarding state simultaneously.

To configure microloop avoidance in a segment routing MPLS network, include the `maximum-labels` and the `maximum-srv6-sids` statements at the `[edit protocols isis spf-options microlooop-avoidance post-convergence-path]` hierarchy level.

When an IPV6 prefix has both SR-MPLS- MLA and SRV6 micro-loop-avoiding paths available, we will prefer the SR-MPLS MLA path. SR-MPLS can provide micro-loop-avoiding paths for ipv4/ipv6 prefixes and SR-labels. Delay specifies the time in milliseconds for which we use the Micro-loop-Avoidance path, before transitioning to SPF path. Note that microloop avoidance is not a replacement for local repair mechanisms such as topology-independent loop-free alternate (TI-LFA), which detects local failure very fast and activates a precomputed loop-free alternative path. Routers that implement micro-loop avoidance compute the micro-loop avoiding path only after receiving the link state update for the event. So, micro-loop avoidance mechanism is not a replacement for local repair mechanisms like TI-LFA

which detect local failure very fast and activate a pre-computed loop-free-alternative path at PFE level. In the above example, if local repair mechanism is not present for the R2R3 failure, there will be lot of traffic loss before R0 can detect the failure(via global convergence) and program a micro-loop avoiding path. Micro-loop avoidance can't avoid traffic loss due to delayed detection of the failure. Micro-loop avoidance will avoid traffic loss due to micro-loops only. Both local-repair mechanisms like TI-LFA and micro-loop avoidance, will have to be enabled on all the nodes in the network to ensure that traffic loss is in milli-seconds range.

To avoid micro-loops, the following process is used:

1. After computing the new path to D, for a predetermined time, R installs an entry for D that steers packets to D via a loop-free SR path. This time should be greater than worst case delay of any router in the network.

2. After the configured time delay, R installs the post-convergence route entry for D, which is without any SIDs.

NOTE: If microloop avoidance is configured for both SRv6 and SR-MPLS, IS-IS prefers to take the SR-MPLS path.

In case the microloop path requires more than the number of SIDs (SR-MPLS or SRv6 SIDs) the platform supports, then only the global convergence primary path is installed.

To configure microloop avoidance in an OSPFv2 segment routing network for both local and remote network events including link down, link-up, and metric-change, configure:

- The maximum number of labels installed for post-convergence paths by including the `maximum-labels` statement at the `[edit protocols ospf spf-options microloop avoidance post-convergence-path]` hierarchy level.
- The time after which temporary post-convergence paths are removed by including the `delay` *`milliseconds`* statement at the `[edit protocols ospf spf-options microloop avoidance post-convergence-path]` hierarchy level.

For effective microloop avoidance, configure this feature on all the nodes in the network.

NOTE: For OSPFv2, Microloop avoidance is supported for IPv4 networks only.

Supported Platforms and Unsupported Features

Junos OS supports microloop avoidance on most platforms that support IS-IS. For details on specific devices and Junos OS releases that support IS-IS micro loop avoidance, see Feature Explorer.

Junos OS does not support the following features in conjunction with microloop avoidance:

- Cannot prevent traffic loss because of slow control plane convergence.
- If shortcuts are available IGP does not provide a microloop avoidance path.
- Microloop avoidance path that needs more than 8 labels is not supported for OSPFv2. The maximum number of labels installed for microloop avoidance path is 8. For the microloop avoidance ECMP path to be usable, the number of labels must be less than or equal to maximum labels.
- Adjacency SIDs are not supported with OSPFv2 microloop avoidance.
- OSPFv2 multi-topology is not supported with microloop avoidance.

RELATED DOCUMENTATION

microloop-avoidance

Configuring Segment Routing Microloop Avoidance in OSPFv2 Networks

IN THIS SECTION

Overview

Microloops are packet forwarding loops that occur in the network following network change events such as link down, link up, or metric change. When a network change event occurs, different routers update their forwarding states at different times. This can lead to packets getting looped between upstream and downstream routers for a transient period, resulting in packet loss, jitter, and out-of-order packets. Microloops can consume the available bandwidth of the links, which impacts the efficient transmission of useful packets.

Microloop avoidance can prevent forwarding of looping packets. The segment routing microloop avoidance detects if microloops are possible following a topology change. When a network change

event is detected, the routes are programmed to take the post-convergence path, that uses a combination of node and adjacency SIDs. This ensures the routers that might not yet have converged do not loop the packets causing microloops. This behavior lasts for a configurable delay. Once the delay timer expires, routes are programmed normally by using node-SID of the destinations.

Requirements

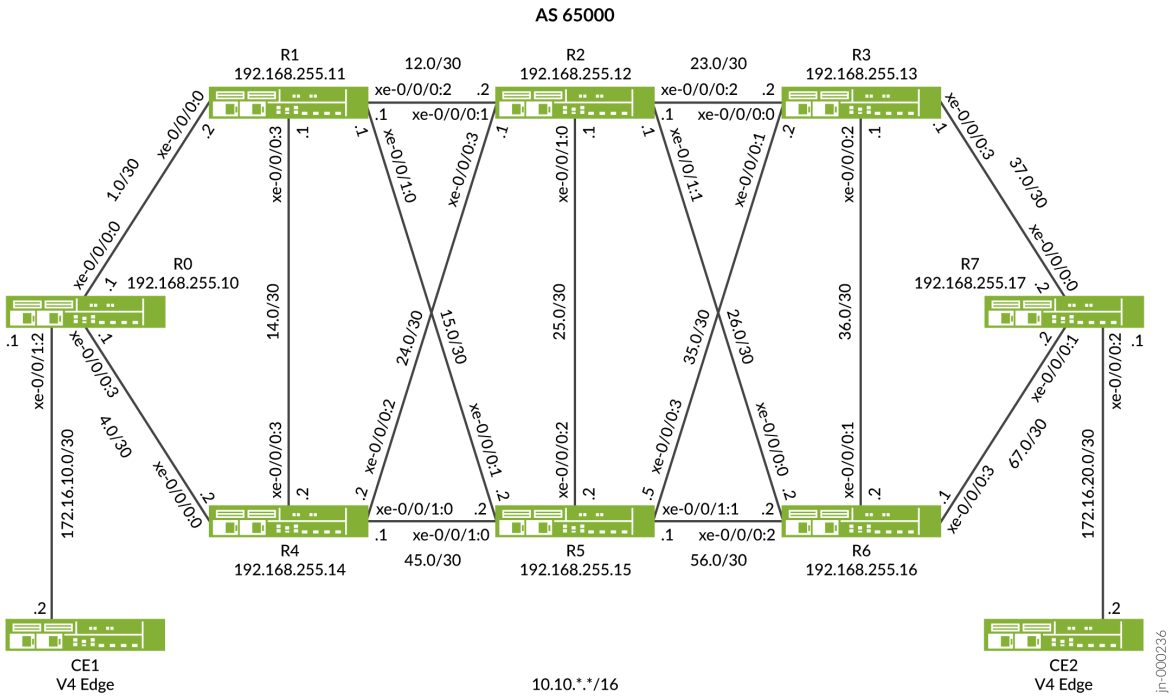
This example uses the following hardware and software components:

- Eight MX Series routers.
- Junos OS Release 22.1R1 or later.

Topology

In Figure 14 on page 110 device R0 and device R7 are the ingress and egress routers that support devices CE1 and CE2. The devices R1, R2, R3, R4, R5, and R6 comprise an IPv4 only provider core network. All the devices belong to the same autonomous system. OSPFv2 is the interior gateway protocol in the core configured to support microloop avoidance. In this example the device R2 is configured as an IPv4 route reflector with IBGP peering sessions to both R0 and R7. No other routers speak BGP in this example. The Device R6 has the firewall filter configured to detect packets with microloops if any following a link down event.

Figure 14: Microloop Avoidance Topology



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Device R0

```
set interfaces xe-0/0/0:0 description To_R1
set interfaces xe-0/0/0:0 unit 0 family inet address 10.10.1.1/30
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces xe-0/0/0:3 description To_R4
set interfaces xe-0/0/0:3 unit 0 family inet address 10.10.4.1/30
set interfaces xe-0/0/0:3 unit 0 family mpls
set interfaces xe-0/0/1:2 description to_CE1
set interfaces xe-0/0/1:2 unit 1 family inet address 172.16.10.2/30
set interfaces xe-0/0/1:2 unit 1 family mpls
set interfaces xe-0/0/1:2 unit 4 family inet address 172.16.11.2/30
set interfaces xe-0/0/1:2 unit 4 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.10/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.10/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1000
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 192.168.255.10
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.10
set protocols bgp group to-RR neighbor 192.168.255.12 family inet unicast
set protocols bgp group to-RR neighbor 192.168.255.12 family inet-vpn unicast per-prefix-label
set protocols mpls traffic-engineering
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf spf-options microloop-avoidance post-convergence-path delay 60000
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing node-segment ipv4-index 0
set protocols ospf source-packet-routing srgb start-label 800000
set protocols ospf source-packet-routing srgb index-range 80000
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 interface-type p2p
```

```
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 post-convergence-lfa node-protection
```

Device R1

```
set interfaces xe-0/0/0:0 description To_R0
set interfaces xe-0/0/0:0 unit 0 family inet address 10.10.1.2/30
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces xe-0/0/0:2 description To_R2
set interfaces xe-0/0/0:2 unit 0 family inet address 10.10.12.1/30
set interfaces xe-0/0/0:2 unit 0 family mpls
set interfaces xe-0/0/0:2 unit 1 family inet address 10.11.12.1/30
set interfaces xe-0/0/0:2 unit 1 family mpls
set interfaces xe-0/0/0:3 description to_R4
set interfaces xe-0/0/0:3 unit 0 family inet address 10.10.14.1/30
set interfaces xe-0/0/0:3 unit 0 family mpls
set interfaces xe-0/0/1:0 description to_R5
set interfaces xe-0/0/1:0 unit 0 family inet address 10.10.15.1/30
set interfaces xe-0/0/1:0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.11/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.11/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1001
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 192.168.255.11
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols mpls traffic-engineering
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf spf-options microloop-avoidance post-convergence-path delay 60000
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf source-packet-routing prefix-segment prefix-sid
```

```
set protocols ospf source-packet-routing node-segment ipv4-index 2
set protocols ospf source-packet-routing srgb start-label 800000
set protocols ospf source-packet-routing srgb index-range 80000
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 post-convergence-lfa
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 post-convergence-lfa
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 post-convergence-lfa
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.1 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.1 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 post-convergence-lfa
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.1 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.1 metric 10
```

Device R2

```
set interfaces xe-0/0/0:1 description To_R1
set interfaces xe-0/0/0:1 unit 0 family inet address 10.10.12.2/30
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces xe-0/0/0:1 unit 1 family inet address 10.11.12.2/30
set interfaces xe-0/0/0:1 unit 1 family inet6
set interfaces xe-0/0/0:1 unit 1 family mpls
set interfaces xe-0/0/0:2 description To_R3
set interfaces xe-0/0/0:2 unit 0 family inet address 10.10.23.1/30
set interfaces xe-0/0/0:2 unit 0 family mpls
set interfaces xe-0/0/0:3 description To_R4
set interfaces xe-0/0/0:3 unit 0 family inet address 10.10.24.1/30
set interfaces xe-0/0/0:3 unit 0 family mpls
set interfaces xe-0/0/1:0 description To_R5
set interfaces xe-0/0/1:0 unit 0 family inet address 10.10.25.1/30
set interfaces xe-0/0/1:0 unit 0 family mpls
set interfaces xe-0/0/1:1 description To_R6
set interfaces xe-0/0/1:1 unit 0 family inet address 10.10.26.1/30
set interfaces xe-0/0/1:1 unit 0 family mpls
```

```
set interfaces lo0 unit 0 family inet address 192.168.255.12/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.12/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1002
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 192.168.255.12
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.12
set protocols bgp group to-RR neighbor 192.168.255.17 family inet unicast
set protocols bgp cluster 192.168.255.12
set protocols mpls traffic-engineering
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf spf-options microloop-avoidance post-convergence-path delay 60000
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing node-segment ipv4-index 4
set protocols ospf source-packet-routing srgb start-label 800000
set protocols ospf source-packet-routing srgb index-range 80000
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/1:1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/1:1.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/1:1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.1 interface-type p2p
```

```
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.1 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.2 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.2 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.3 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.3 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.4 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.4 metric 10
```

Device R3

```
set interfaces xe-0/0/0:0 description To_R2
set interfaces xe-0/0/0:0 unit 0 family inet address 10.10.23.2/30
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces xe-0/0/0:1 description To_R5
set interfaces xe-0/0/0:1 unit 0 family inet address 10.10.35.2/30
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces xe-0/0/0:2 description To_R6
set interfaces xe-0/0/0:2 unit 0 family inet address 10.10.36.1/30
set interfaces xe-0/0/0:2 unit 0 family mpls
set interfaces xe-0/0/0:3 description To_R7
set interfaces xe-0/0/0:3 unit 0 family inet address 10.10.37.1/30
set interfaces xe-0/0/0:3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.13/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.13/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1003
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 192.168.255.13
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols mpls traffic-engineering
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf spf-options microloop-avoidance post-convergence-path delay 60000
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf source-packet-routing prefix-segment prefix-sid
```

```
set protocols ospf source-packet-routing node-segment ipv4-index 6
set protocols ospf source-packet-routing srgb start-label 800000
set protocols ospf source-packet-routing srgb index-range 80000
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 post-convergence-lfa node-protection
```

Device R4

```
set interfaces xe-0/0/0:0 description To_R0
set interfaces xe-0/0/0:0 unit 0 family inet address 10.10.4.2/30
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces xe-0/0/0:2 description To_R2
set interfaces xe-0/0/0:2 unit 0 family inet address 10.10.24.2/30
set interfaces xe-0/0/0:2 unit 0 family mpls
set interfaces xe-0/0/0:3 description To_R1
set interfaces xe-0/0/0:3 unit 0 family inet address 10.10.14.2/30
set interfaces xe-0/0/0:3 unit 0 family mpls
set interfaces xe-0/0/1:0 description To_R5
set interfaces xe-0/0/1:0 unit 0 family inet address 10.10.45.1/30
set interfaces xe-0/0/1:0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.14/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.14/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1004
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 192.168.255.14
set routing-options forwarding-table export pplb
set routing-options autonomous-system 65000
```

```
set protocols mpls traffic-engineering
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf spf-options microloop-avoidance post-convergence-path delay 60000
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing node-segment ipv4-index 8
set protocols ospf source-packet-routing srgb start-label 800000
set protocols ospf source-packet-routing srgb index-range 80000
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 post-convergence-lfa node-protection
```

Device R5

```
set interfaces xe-0/0/0:1 description To_R1
set interfaces xe-0/0/0:1 unit 0 family inet address 10.10.15.2/30
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces xe-0/0/0:2 description To_R2
set interfaces xe-0/0/0:2 unit 0 family inet address 10.10.25.2/30
set interfaces xe-0/0/0:2 unit 0 family mpls
set interfaces xe-0/0/0:3 description To_R3
set interfaces xe-0/0/0:3 unit 0 family inet address 10.10.35.2/30
set interfaces xe-0/0/0:3 unit 0 family mpls
set interfaces xe-0/0/1:0 description To_R4
set interfaces xe-0/0/1:0 unit 0 family inet address 10.10.45.2/30
set interfaces xe-0/0/1:0 unit 0 family mpls
set interfaces xe-0/0/1:1 description To_R6
```

```
set interfaces xe-0/0/1:1 unit 0 family inet address 10.10.56.1/30
set interfaces xe-0/0/1:1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.15/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.15/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1005
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 192.168.255.15
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols mpls traffic-engineering
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf spf-options microloop-avoidance post-convergence-path delay 60000
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf source-packet-routing node-segment ipv4-index 10
set protocols ospf source-packet-routing srgb start-label 800000
set protocols ospf source-packet-routing srgb index-range 80000
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/1:1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/1:1.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/1:1.0 post-convergence-lfa node-protection
```

Device R6

```
set interfaces xe-0/0/0:0 description To_R2
set interfaces xe-0/0/0:0 unit 0 family inet address 10.10.26.2/30
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces xe-0/0/0:1 description To_R3
set interfaces xe-0/0/0:1 unit 0 family inet address 10.10.36.2/30
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces xe-0/0/0:2 description To_R5
set interfaces xe-0/0/0:2 unit 0 family inet filter output v4filter
set interfaces xe-0/0/0:2 unit 0 family inet address 10.10.56.2/30
set interfaces xe-0/0/0:2 unit 0 family mpls filter output mplsfilter
set interfaces xe-0/0/0:3 description To_R7
set interfaces xe-0/0/0:3 unit 0 family inet address 10.10.67.1/30
set interfaces xe-0/0/0:3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.16/32
set interfaces lo0 unit 0 family inet address 192.168.255.61/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.16/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1006
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set policy-options policy-statement prefix-sid term 2 from route-filter 192.168.255.61/32 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 1106
set policy-options policy-statement prefix-sid term 2 then accept
set firewall family inet filter v4filter term t1 from destination-address 8.3.0.0/16
set firewall family inet filter v4filter term t1 then accept
set firewall family inet filter v4filter term t6 then accept
set firewall family mpls filter mplsfilter term t1 from ip-version ipv4 destination-address
10.8.0.1/16
set firewall family mpls filter mplsfilter term t1 then count v4sr-nsid-cnt
set firewall family mpls filter mplsfilter term t1 then accept
set firewall family mpls filter mplsfilter term t2 from ip-version ipv4 destination-address
10.9.0.1/16
set firewall family mpls filter mplsfilter term t2 then count v4sr-psid-cnt
set firewall family mpls filter mplsfilter term t2 then accept
set firewall family mpls filter mplsfilter term t3 then accept
set firewall family mpls filter mplsfilter term t4 then accept
set firewall family mpls filter mplsfilter term t6 then accept
set routing-options router-id 192.168.255.16
```

```
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols mpls traffic-engineering
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf spf-options microloop-avoidance post-convergence-path delay 60000
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing node-segment ipv4-index 12
set protocols ospf source-packet-routing srgb start-label 800000
set protocols ospf source-packet-routing srgb index-range 80000
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 metric 110
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 metric 100
set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 post-convergence-lfa node-protection
```

Device R7

```
set interfaces xe-0/0/0:0 description To_R3
set interfaces xe-0/0/0:0 unit 0 family inet address 10.10.37.2/24
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces xe-0/0/0:1 description To_R6
set interfaces xe-0/0/0:1 unit 0 family inet address 10.10.67.2/30
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces xe-0/0/0:2 description to_CE2
set interfaces xe-0/0/0:2 unit 4 family inet address 172.16.20.1/30
set interfaces xe-0/0/0:2 unit 4 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.17/32
set interfaces lo0 unit 0 family inet address 192.168.255.71/32
```

```
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement payload_9 term 1 from route-filter 10.7.0.1/16 orlonger
set policy-options policy-statement payload_9 term 1 then next-hop 192.168.255.17
set policy-options policy-statement payload_9 term 1 then accept
set policy-options policy-statement payload_9 term 2 from route-filter 10.8.0.1/16 orlonger
set policy-options policy-statement payload_9 term 2 then next-hop 192.168.255.17
set policy-options policy-statement payload_9 term 2 then accept
set policy-options policy-statement payload_9 term 3 from route-filter 8.2.0.0/16 orlonger
set policy-options policy-statement payload_9 term 3 then next-hop 192.168.255.71
set policy-options policy-statement payload_9 term 4 then reject
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.17/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1007
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set policy-options policy-statement prefix-sid term 2 from route-filter 192.168.255.71/32 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 1107
set policy-options policy-statement prefix-sid term 2 then accept
set policy-options policy-statement v4stat term 1 from protocol static
set policy-options policy-statement v4stat term 1 from route-filter 100.100.100.1/32 orlonger
set policy-options policy-statement v4stat term 1 then accept
set policy-options policy-statement v4_prefixes term 1 from route-filter 8.3.0.0/16 orlonger
set policy-options policy-statement v4_prefixes term 1 then accept
set policy-options policy-statement v4_prefixes term 3 then reject
set routing-options rib inet.0 static route 100.100.100.1/32 receive
set routing-options router-id 192.168.255.17
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.17
set protocols bgp group to-RR neighbor 192.168.255.12 family inet unicast
set protocols bgp group to-RR neighbor 192.168.255.12 export payload_9
set protocols bgp group to-CE1 type external
set protocols bgp group to-CE1 local-address 172.16.20.1
set protocols bgp group to-CE1 neighbor 172.16.20.2 family inet unicast
set protocols bgp group to-CE1 neighbor 172.16.20.2 peer-as 700
set protocols bgp group to-CE1 neighbor 172.16.20.2 local-as 100
set protocols mpls traffic-engineering
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
```

```
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing node-segment ipv4-index 14
set protocols ospf source-packet-routing srgb start-label 800000
set protocols ospf source-packet-routing srgb index-range 80000
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0 post-convergence-lfa node-protection
```

Configuring Device R0

Step-by-Step Procedure

To configure segment routing microloop avoidance path in an OSPFv2 network, perform the following steps on the R0 device:

1. Configure the device interfaces to enable IP and MPLS transport.

```
[edit]
user@R0#set interfaces xe-0/0/0:0 description To_R1
user@R0#set interfaces xe-0/0/0:0 unit 0 family inet address 10.10.1.1/30
uesr@R0#set interfaces xe-0/0/0:0 unit 0 family mpls
user@R0#set interfaces xe-0/0/0:3 description To_R4
user@R0#set interfaces xe-0/0/0:3 unit 0 family inet address 10.10.4.1/30
uesr@R0#set interfaces xe-0/0/0:3 unit 0 family mpls
user@R0#set interfaces xe-0/0/1:2 description to_CE1
user@R0#set interfaces xe-0/0/1:2 unit 1 family inet address 172.16.10.2/30
user@R0#set interfaces xe-0/0/1:2 unit 1 family mpls
```

2. Configure the loopback interface (lo0) addresses that is used as router ID for OSPF sessions.

```
[edit]
user@R0#set interfaces lo0 unit 0 family inet address 192.168.255.10/32
user@R0#set interfaces lo0 unit 0 family inet address 192.168.255.18/32
```

3. Configure the router ID and autonomous system (AS) number to propagate routing information within a set of routing devices that belong to the same AS.

```
[edit]
user@R0#set routing-options router-id 192.168.255.10
user@R0#set routing-options autonomous-system 65000
```

4. Define a policy to load balance packets and apply the per-packet policy to enable load balancing of traffic.

```
[edit]
user@R0#set policy-options policy-statement pplb then load-balance per-packet
user@R0#set routing-options forwarding-table export pplb
```

5. Configure R0 to advertise the loopback address. The `prefix-segment index` option sets the base label for each router's loopback. In this example the base index is set to reflect| the router number. As a result, R0 uses 1000.

```
[edit]
user@R0#set policy-options policy-statement prefix-sid term 1 from route-filter
192.168.255.10/32 exact
user@R0#set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1000
user@R0#set policy-options policy-statement prefix-sid term 1 then prefix-segment node-
segment
user@R0#set policy-options policy-statement prefix-sid term 1 then accept
```

6. Configure MPLS on all interfaces excluding the management interface. Also enable traffic engineering.

```
[edit]
user@R0#set protocols mpls interface all
user@R0#set protocols mpls interface fxp0.0 disable
user@R0#set protocols mpls traffic-engineering
```

7. Configure the MPLS label range to assign static labels for the links.

```
[edit]
user@R0#set protocols mpls label-range static-label-range 60001 100000
```

8. Configure BGP peering between R0 and the route reflector R2. Configure the unicast network layer reachability information (NRLI) to allocate a unique label for each prefix on the devices.

```
[edit]
user@R0#set protocols bgp group to-RR type internal
user@R0#set protocols bgp group to-RR local-address 192.168.255.10
user@R0#set protocols bgp group to-RR neighbor 192.168.255.12 family inet unicast
user@R0#set protocols bgp group to-RR neighbor 192.168.255.12 family inet-vpn unicast per-
prefix-label
```

9. Configure TI-LFA to enable protection against link and node failures. SR using TI-LFA provides faster restoration of network connectivity by routing the traffic instantly to a backup or an alternate path if the primary path fails or becomes unavailable.

```
[edit]
user@host#set protocols ospf backup-spf-options use-source-packet-routing
```

10. Configure backup shortest path first (SPF) attributes such as maximum equal-cost multipath (ECMP) as 8 and maximum number of labels as 5 for TI-LFA for the OSPFv2 protocol.

```
[edit]
user@host#set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
user@host#set protocols ospf backup-spf-options use-post-convergence-lfa maximum-backup-
paths 8
```

11. Configure prefix segment attributes, the start label and the index range for segment routing global blocks (SRGBs) in SPRING for the OSPFv2 protocol.

```
[edit]
user@host#set protocols ospf source-packet-routing prefix-segment prefix-sid
user@host#set protocols ospf source-packet-routing node-segment ipv4-index 0
```

```
user@host#set protocols ospf source-packet-routing srgb start-label 800000
user@host#set protocols ospf source-packet-routing srgb index-range 80000
```

12. Configure the loopback interface as passive to ensure the protocols do not run over the loopback interface and that the loopback interface is advertised correctly throughout the network.

```
[edit]
user@host#set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

13. Configure OSPF area 0 on the point-to-point interface of the device R0.

```
[edit]
user@host#set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 interface-type p2p
user@host#set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 metric 10
user@host#set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 interface-type p2p
user@host#set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 metric 10
```

14. Configure the computation and installation of a backup path that follows the post-convergence path on the given area and interface for the OSPFv2 protocol. Also enable node-link protection on the these interfaces that follow post-convergence path.

```
[edit]
user@host#set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0 post-convergence-lfa node-
protection
user@host#set protocols ospf area 0.0.0.0 interface xe-0/0/0:3.0 post-convergence-lfa node-
protection
```

15. Configure microloop avoidance that temporarily installs a post-convergence path for routes potentially affected by microloops and specify a delay time period of 60000 milliseconds for the OSPFv2 protocol. The temporary path reverts to the node SIDs of the destination after the delay timer expires.

```
[edit]
user@host#set protocols ospf spf-options microloop-avoidance post-convergence-path delay
60000
```

Results

Check the results of the configuration:

```
interfaces {
    xe-0/0/0:0 {
        description To_R1;
        unit 0 {
            family inet {
                address 10.10.1.1/30;
            }
            family mpls;
        }
    }
    xe-0/0/0:3 {
        description To_R4;
        unit 0 {
            family inet {
                address 10.10.4.1/30;
            }
            family mpls;
        }
    }
    xe-0/0/1:2 {
        description to_CE1;
        unit 1 {
            family inet {
                address 172.16.10.2/30;
            }
            family mpls;
            }
        }
  }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.255.10/32;
                address 192.168.255.18/32;
            }
            family mpls;
        }
    }
}
policy-options {
```

```
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
    policy-statement prefix-sid {
        term 1 {
            from {
                route-filter 192.168.255.10/32 exact;
            }
            then {
                prefix-segment {
                    index 1000;
                    node-segment;
                }
                accept;
            }
        }
        term 2 {
            from {
                route-filter 192.168.255.18/32 exact;
            }
            then {
                prefix-segment {
                    index 1100;
                }
                accept;
            }
        }
    }
}
routing-options {
    router-id 192.168.255.10;
    autonomous-system 100;
    forwarding-table {
        export pplb;
    }
}
protocols {
    bgp {
        group to-RR {
            type internal;
            local-address 192.168.255.10;
```

```
            neighbor 192.168.255.12 {
                family inet {
                    unicast;
                }
                family inet-vpn {
                    unicast {
                        per-prefix-label;
                    }
                }
            }
        }
    }
    mpls {
        traffic-engineering;
        label-range {
            static-label-range 60001 100000;
        }
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    ospf {
        spf-options {
            microloop-avoidance {
                post-convergence-path {
                    delay 60000;
                }
            }
        }
        backup-spf-options {
            use-post-convergence-lfa {
                maximum-labels 5;
                maximum-backup-paths 8;
            }
            use-source-packet-routing;
        }
        source-packet-routing {
            prefix-segment prefix-sid;
            node-segment ipv4-index 0;
            srgb start-label 800000 index-range 80000;
        }
        area 0.0.0.0 {
```

```
            interface lo0.0 {
                passive;
            }
            interface xe-0/0/0:0.0 {
                interface-type p2p;
                metric 10;
                post-convergence-lfa;
            }
            interface xe-0/0/0:3.0 {
                interface-type p2p;
                metric 10;
                post-convergence-lfa;
            }
        }
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

The following section explains microloop avoidance for a link down event.

Verify Connectivity Between R0 and R7 Before the Link is Disabled Between R0 and R1

Purpose

Verify that the Device R0 can reach the destinations on Device R7.

Action

From operational mode, run the **ping** command on the device R0.

```
user@R0>ping 192.168.255.17
PING 192.168.255.17 (192.168.255.17): 56 data bytes
64 bytes from 192.168.255.17: icmp_seq=0 ttl=61 time=41.493 ms
64 bytes from 192.168.255.17: icmp_seq=1 ttl=61 time=57.242 ms
64 bytes from 192.168.255.17: icmp_seq=2 ttl=61 time=44.977 ms
64 bytes from 192.168.255.17: icmp_seq=3 ttl=61 time=202.092 ms
64 bytes from 192.168.255.17: icmp_seq=4 ttl=61 time=60.495 ms
64 bytes from 192.168.255.17: icmp_seq=5 ttl=61 time=39.396 ms
64 bytes from 192.168.255.17: icmp_seq=6 ttl=61 time=79.993 ms
64 bytes from 192.168.255.17: icmp_seq=7 ttl=61 time=78.741 ms
8 packets transmitted, 8 received, 0% packet loss, time 7007ms
rtt min/avg/max/mdev = 38.194/47.998/60.879/8.727 ms
```

Meaning

These results confirm that the device R0 can reach device R7 in the OSPFv2 network.

Verify Disabling the Link Between R0 and R1

Purpose

To verify disabling the link between R0 and R1 on the device R0

Action

From configuration mode, run the **disable interface** command on the device R0

```
user@R0#disble interface xe-0/0/0:0
```

To verify the link is disabled, from operational mode, run the **show interfaces** command on the device R0

```
user@R0>show interfaces xe-0/0/0:0
Physical interface: xe-0/0/0:0, Administratively down, Physical link is Down
  Interface index: 149, SNMP ifIndex: 527
  Description: To_R1_1
  Link-level type: Ethernet, MTU: 1518, MRU: 1526, LAN-PHY mode, Speed: 10Gbps, BPDU Error:
None, Loop Detect PDU Error: None, MAC-REWRITE Error: None, Loopback: None, Source filtering:
Disabled, Flow control: Enabled, Speed Configuration: Auto
  Pad to minimum frame size: Disabled
  Device flags   : Present Running Down
  Interface flags: Hardware-Down Down SNMP-Traps Internal: 0x4000
  CoS queues     : 8 supported, 8 maximum usable queues
  Schedulers     : 0
  Current address: 2c:6b:f5:42:fe:00, Hardware address: 2c:6b:f5:42:fe:00
  Last flapped   : 2022-02-15 09:53:51 PST (00:00:10 ago)
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
  Active alarms  : None
  Active defects : None
  PCS statistics                      Seconds
    Bit errors                             0
    Errored blocks                         0
  Link Degrade :
    Link Monitoring                    :  Disable
  Interface transmit statistics: Disabled
```

Meaning

The output indicates the physical link between R0 and R1 is disabled and is administratively down.

Verify Microloop-avoidance Path Installed for the Destination After the Link is Disabled

Purpose

Verify microloop-avoidance path installed for the destination routes R7 from R0 when the link is disabled between R0 and R1 by verifying routes in the inet.3 table and route label details in the mpls.0 table.

Action

From operational mode, run the **show route table inet.3** command on the device R0.

```
user@R0>show route table inet.3 192.168.255.17/32
inet.3: 25 destinations, 26 routes (25 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.255.17/32         *[L-OSPF/10/5] 00:00:31, metric 130
                    >  to 192.168.255.14 via xe-0/0/0:3, Push 16, Push 801006(top)
```

From operational mode, run the **show route label *label value* protocol ospf extensive** command on the device R0.

```
user@R0>show route label 801007 protocol ospf extensive
mpls.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
801007 (1 entry, 1 announced)
TSI:
KRT in-kernel 801007 /52 -> {Swap 16, Push 801006 (top)}
        *L-OSPF Preference: 10/5
                Next hop type: Router, Next hop index: 649
                Address: 0x7a1ed58
                Next-hop reference count: 4, key opaque handle: 0x0
                Next hop: 10.10.4.2 via xe-0/0/0:3.0 weight 0x1, selected
                Label operation: Swap 16, Push 801006(top)
                Load balance label: Label 16: None; Label 801006: None
                Label element ptr: 0x8fd6ed0
                Label parent element ptr: 0x0
                Label element references: 1
                Label element child references: 0
                Label element lsp id: 0
                Session Id: 321
                State: <Active Int>
                Local AS:   100
                Age: 2:55:13    Metric: 130
                Validation State: unverified
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (1): 1-KRT
```

```
                AS path: I
                Thread: junos-main
```

Meaning

The output indicates that when the link between R0 and R1 goes down, the microloop-avoidance path is installed for R7 from R0 through R4 until the delay timer expires.

Verify Packets With Microloops

Purpose

Verify packets with microloops by using firewall counter information

Action

From operational mode, run the **show firewall** command on the device R6.

```
user@R6>show firewall
Filter: mplsfilter
Counters:
Name                                                Bytes              Packets
v4sr-nsid-cnt                                           0                    0
v4sr-psid-cnt                                           0                    0
```

Meaning

The output displays the mplsfilter configured on the device R6 to display microloops if there are any. The value 0 indicates there are no packets with microloops.

Verify Microloop-avoidance Path Changes to Post-convergence- path After the Delay Timer Expires

Purpose

Verify microloop-avoidance path installed for the destination routes R7 from R0 changes to post-convergence-path after the delay timer 60000 ms expires.

Action

From operational mode, run the **show route table inet.3** command on the device R0.

```
user@R0>show route table inet.3 192.168.255.17/32
inet.3: 25 destinations, 26 routes (25 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.255.17/32         *[L-OSPF/10/5] 00:00:31, metric 130
                     >  to 192.168.255.14 via xe-0/0/0:3, Push 801007
```

From operational mode, run the **show route label *label value* protocol ospf extensive** command on the device R0.

```
user@R0>show route label 801007 protocol ospf extensive
mpls.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
801007 (1 entry, 1 announced)
TSI:
KRT in-kernel 801007 /52 -> {Swap 801007}
        *L-OSPF Preference: 10/5
                Next hop type: Router, Next hop index: 615
                Address: 0x7a1c400
                Next-hop reference count: 4, key opaque handle: 0x0
                Next hop: 10.10.4.2 via xe-0/0/0:3.0 weight 0x1, selected
                Label operation: Swap 801007
                Load balance label: Label 801007: None;
                Label element ptr: 0x8fd6458
                Label parent element ptr: 0x0
                Label element references: 1
                Label element child references: 0
                Label element lsp id: 0
                Session Id: 321
                State: <Active Int>
                Local AS:   100
                Age: 2:55:13    Metric: 130
                Validation State: unverified
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (1): 1-KRT
                AS path: I
                Thread: junos-main
```

Meaning

The output indicates that the microloop-avoidance path is changed to post-convergence-path after the delay timer expires.

Verify Connectivity Between R0 and R7

Purpose

Verify that the Device R0 can reach the destinations on Device R7.

Action

From operational mode, run the **ping** command on the device R0.

```
user@R0>ping 192.168.255.17
PING 192.168.255.17 (192.168.255.17): 56 data bytes
64 bytes from 192.168.255.17: icmp_seq=0 ttl=61 time=41.493 ms
64 bytes from 192.168.255.17: icmp_seq=1 ttl=61 time=57.242 ms
64 bytes from 192.168.255.17: icmp_seq=2 ttl=61 time=44.977 ms
64 bytes from 192.168.255.17: icmp_seq=3 ttl=61 time=202.092 ms
64 bytes from 192.168.255.17: icmp_seq=4 ttl=61 time=60.495 ms
64 bytes from 192.168.255.17: icmp_seq=5 ttl=61 time=39.396 ms
64 bytes from 192.168.255.17: icmp_seq=6 ttl=61 time=79.993 ms
64 bytes from 192.168.255.17: icmp_seq=7 ttl=61 time=78.741 ms
8 packets transmitted, 8 received, 0% packet loss, time 7007ms
rtt min/avg/max/mdev = 38.194/47.998/60.879/8.727 ms
```

Meaning

These results confirm that the device R0 can reach device R7 in the OSPFv2 network and that the traffic flows with 0% packet loss in case of link down because of the microloop-avoidance path configured.

Verify the Path Changes to Microloop-avoidance Path After the Link is Enabled

Purpose

Verify the path changes to microloop-avoidance path for the destination when the link is enabled between R0 and R1.

Action

From operational mode, run the **show route table inet.3** command on the device R0.

```
user@R0>show route table inet.3 192.168.255.17/32
inet.3: 26 destinations, 27 routes (26 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.255.17/32         *[L-OSPF/10/5] 00:02:05, metric 40
                    >  to 192.168.255.11 via xe-0/0/0:0, Push 801007
                       to 192.168.255.14 via xe-0/0/0:3, Push 16, Push 801006(top)
```

From operational mode, run the **show route label *label value* protocol ospf extensive** command on the device R0.

```
user@R0>show route label 801007 protocol ospf extensive
mpls.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
801007 (1 entry, 1 announced)
TSI:
KRT in-kernel 801007 /52 -> {list:Swap 801007, Swap 16, Push 801006(top)}
        *L-OSPF Preference: 10/5
                Next hop type: Router, Next hop index: 615
                Address: 0x79329ac
                Next-hop reference count: 3, key opaque handle: 0x0
                Next hop: 10.10.4.2 via xe-0/0/0:3.0 weight 0x1, selected
                Label operation: Push 801007
                Load balance label: Label 801007: None;
                Label element ptr: 0x8fd6458
                Label parent element ptr: 0x0
                Label element references: 1
                Label element child references: 0
                Label element lsp id: 0
                Session Id: 0
                Next hop: 10.10.1.2 via xe-0/0/0:0.0 weight 0xf000, selected
                Label operation: Swap 16, Push 801006(top)
                Load balance label: Label 16: None; Label 801006: None;
                Label element ptr: 0x8fd8e60
                Label parent element ptr: 0x0
                Label element references: 1
                Label element child references: 0
                Label element lsp id: 0
```

```
Session Id: 0
State: <Active Int>
Local AS:   100
Age: 2:55:13    Metric: 40
Validation State: unverified
Area: 0.0.0.0
Task: OSPF
Announcement bits (1): 1-KRT
AS path: I
Thread: junos-main
```

Meaning

The output displays the routes to the destination R7 from R0 which includes microloop-avoidance path and the post-convergence path after the link is enabled between R0 and R7.

Migration to Segment Routing

IN THIS SECTION

Mapping Client and Server for Segment Routing to LDP Interoperability

IN THIS SECTION

Segment routing mapping server and client support enables interoperability between network islands that run LDP and segment routing (SR or SPRING). This interoperability is useful during a migration from LDP to SR. During the transition there can be islands (or domains) with devices that support either only LDP, or only segment routing. For these devices to interwork the LDP segment routing mapping server (SRMS) and segment routing mapping client (SRMC) functionality is required. You enable these server and client functions on a device in the segment routing network.

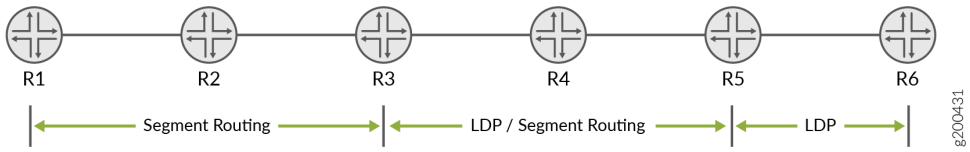
SR mapping server and client functionality is supported with either OSPF or ISIS.

Overview of Segment Routing to LDP Interoperability

Figure 15 on page 138 shows a simple LDP network topology to illustrate how interoperability of segment routing devices with LDP devices works. Keep in mind that both OSPF and ISIS are supported, so for now we'll keep things agnostic with regard to the IGP. The sample topology has six devices, R1 through R6, in a network that is undergoing a migration from LDP to segment routing.

In the topology, devices R1, R2, and R3 are configured for segment routing only. Devices R5 and R6 are part of a legacy LDP domain and do not currently support SR. Device R4 supports both LDP and segment routing. The loopback addresses of all devices are shown. These loopbacks are advertised as egress FECs in the LDP domain and as SR node IDs in the SR domain. Interoperability is based on mapping a LDP FEC into a SR node ID, and vice versa.

Figure 15: Sample Segment Routing to LDP Interoperation Topology



For R1 to interwork with R6, both an LDP segment routing mapping server (SRMS) and a segment routing mapping client (SRMC) are needed. Its easier to understand the role of the SRMS and SRMC by looking at the traffic flow in a unidirectional manner. Based on Figure 15 on page 138, we'll say that traffic flowing from left to right originates in the SR domain and terminates in the LDP domain. In like fashion, traffic that flows from right to left originates in the LDP domain and terminates in the SR domain.

The SRMS provides the information needed to stitch traffic in the left to right direction. The SRMC provides mapping for traffic that flows from right to left.

- Left to right Traffic Flow: The Segment Routing Mapping Server

  The SRMS facilitates LSP stitching between the SR and LDP domains. The server maps LDP FECs into SR node IDs. You configure the LDP FECs to be mapped under the `[edit routing-options source-packet-routing]` hierarchy level. Normally you need to map all LDP node loopback addresses for full

connectivity. As shown below, you can map contiguous prefixes in a single range statement. If the LDP node loopbacks are not contiguous you need to define multiple mapping statements.

You apply the SRMS mapping configuration under the `[edit protocols ospf]` or `[edit protocols isis]` hierarchy level. This choice depends on which IGP is being used. Note that both the SR and LDP nodes share a common, single area/level, IGP routing domain.

The SRMS generates an extended prefix list LSA (or LSP in the case of ISIS). The information in this LSA allows the SR nodes to map LDP prefixes (FECs) to SR Node IDs. The mapped routes for the LDP prefixes are installed in the `inet.3` and `mpls.0` routing tables of the SR nodes to facilitate LSP ingress and stitching operations for traffic in the left to right direction.

The extended LSA (or LSP) is flooded throughout the (single) IGP area. This means you are free to place the SRMS configuration on any router in the SR domain. The SRMS node does not have to run LDP.

- Right to Left Traffic Flow: The Segment Routing Mapping Client

  To interoperate in the right to left direction, that is, from the LDP island to the SR island, you simply enable segment routing mapping client functionality on a node that speaks both SR and LDP. In our example that is R4. You activate SRMC functionality with the `mapping-client` statement at the `[edit protocols ldp]` hierarchy.

  The SRMC configuration automatically activates an LDP egress policy to advertise the SR domain's node and prefix SIDs as LDP egress FECs. This provides the LDP nodes with LSP reachability to the nodes in the SR domain.

- The SRMC function must be configured on a router that attaches to both the SR and LSP domains. If desired, the same node can also function as the SRMS.

Segment Routing to LDP Interoperability Using OSPF

Refer to Figure 15 on page 138, assume that device R2 (in the segment routing network) is the SRMS.

1. Define the SRMS function:

```
[edit routing-options source-packet-routing ]
user@R2# set mapping-server-entry ospf-mapping-server prefix-segment-range ldp-lo0s start-
prefix 192.168.0.5
user@R2# set mapping-server-entry ospf-mapping-server prefix-segment-range ldp-lo0s start-
index 1000
user@R2# set mapping-server-entry ospf-mapping-server prefix-segment-range ldp-lo0s size 2
```

This configuration creates a mapping block for both the LDP device loopback addresses in the sample topology. The initial Segment ID (SID) index mapped to R5's loopback is `1000`. Specifying size `2` results in SID index 10001 being mapped to R6's loopback address.

NOTE: The IP address used as the `start-prefix` is a loopback address of a device in the LDP network (R5, in this example). For full connectivity you must map all the loopback addresses of the LDP routers into the SR domain. If the loopback addresses are contiguous, you can do this with a single `prefix-segment-range` statement. Non-contiguous loopbacks requires definition of multiple prefix mapping statements.

Our example uses contiguous loopbacks so a single `prefix-segment-range` is shown above. Here's an example of multiple mappings to support the case of two LDP nodes with non-contiguous loopback addressing:

```
[edit routing-options source-packet-routing]
show
    mapping-server-entry map-server-name {
        prefix-segment-range lo1 {
            start-prefix 192.168.0.5/32;
            start-index 1000;
            size 1;
        }
        prefix-segment-range lo2 {
            start-prefix 192.168.0.10/32;
            start-index 2000;
            size 1;
        }
    }
}
```

2. Next, configure OSPF support for the extended LSA used to flood the mapped prefixes.

```
[edit protocols]
user@R2# set ospf source-packet-routing mapping-server ospf-mapping-server
```

Once the mapping server configuration is committed on device R2, the extended prefix range TLV is flooded across the OSPF area. The devices capable of segment routing (R1, R2, and R3) install OSPF segment routing routes for the specified loopback address (R5 and R6 in this example), with a segment ID (SID) index. The SID index is also updated in the `mpls.0` routing table by the segment routing devices.

3. Enable SRMC functionality. For our sample topology you must enable SRMC functionality on R4.

```
[edit protocols]
user@R4# set ldp sr-mapping-client
```

   Once the mapping client configuration is committed on device R4, the SR node IDs and label blocks are advertised as egress FECs to router R5, which then re-advertises them to R6.

Support for stitching segment routing and LDP next-hops with OSPF began in Junos OS 19.1R1.

Unsupported Features and Functionality for Segment Routing interoperability with LDP using OSPF

- IPv6 prefixes are not supported.
- Flooding of the OSPF Extended Prefix Opaque LSA across AS boundaries (inter-AS) is not supported.
- Inter-area LDP mapping server functionality is not supported.
- ABR functionality of Extended Prefix Opaque LSA is not supported.
- ASBR functionality of Extended Prefix Opaque LSA is not supported.
- The segment routing mapping server Preference TLV is not supported.

Interoperability of Segment Routing with LDP Using ISIS

Refer to Figure 15 on page 138, assume that device R2 (in the segment routing network) is the SRMS. The following configuration is added for the mapping function:

1. Define the SRMS function:

```
[edit routing-options source-packet-routing ]
user@R2# set mapping-server-entry isis-mapping-server prefix-segment-range ldp-lo0s start-
prefix 192.168.0.5
user@R2# set mapping-server-entry isis-mapping-server prefix-segment-range ldp-lo0s start-
index 1000
user@R2# set mapping-server-entry isis-mapping-server prefix-segment-range ldp-lo0s size 2
```

   This configuration creates a mapping block for both the LDP device loopback addresses in the sample topology. The initial segment ID (SID) index mapped to R5's loopback is `1000`. Specifying size `2` results in SID index 10001 being mapped to R6's loopback address.

NOTE: The IP address used as the `start-prefix` is a loopback address of a device in the LDP network (R5, in this example). For full connectivity you must map all the loopback addresses of the LDP routers in the SR domain. If the loopback addresses are contiguous, you can do this with a `prefix-segment-range` statement. Non-contiguous loopbacks require the definition of multiple mapping statements.

Our example uses contiguous loopbacks so a single `prefix-segment-range` is shown above. Here is an example of prefix mappings to handle the case of two LDP routers with non-contiguous loopback addressing:

```
[edit routing-options source-packet-routing]
show
    mapping-server-entry map-server-name {
        prefix-segment-range lo1 {
            start-prefix 192.168.0.5/32;
            start-index 1000;
            size 1;
        }
        prefix-segment-range lo2 {
            start-prefix 192.168.0.10/32;
            start-index 2000;
            size 1;
        }
    }
}
```

2. Next, configure ISIS support for the extended LSP used to flood the mapped prefixes.

```
[edit protocols]
user@R2# set isis source-packet-routing mapping-server isis-mapping-server
```

Once the mapping server configuration is committed on device R2, the extended prefix range TLV is flooded across the OSPF area. The devices capable of segment routing (R1, R2, and R3) install ISIS segment routing routes for the specified loopback address (R5 and R6 in this example), with a segment ID (SID) index. The SID index is also updated in the `mpls.0` routing table by the segment routing devices.

3. Enable SRMC functionality. For our sample topology you must enable SRMC functionality on R4.

```
[edit protocols]
user@R4# set ldp sr-mapping-client
```

   Once the mapping client configuration is committed on device R4, the SR node IDs and label blocks are advertised as egress FECs to router R5, and from there on to R6.

Support for stitching segment routing and LDP next-hops with ISIS began in Junos OS 17.4R1.

Unsupported Features and Functionality for Interoperability of Segment Routing with LDP using ISIS

- Penultimate-hop popping behavior for label binding TLV is not supported.
- Advertising of range of prefixes in label binding TLV is not supported.
- Segment Routing Conflict Resolution is not supported.
- LDP traffic statistics does not work.
- Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) is not supported.
- ISIS inter-level is not supported.
- RFC 7794, *IS-IS Prefix Attributes for Extended IPv4* is not supported.
- Redistributing LDP route as a prefix-sid at the stitching node is not supported.

How to Enable Strict SPF SIDs and IGP Shortcut

IN THIS SECTION

Understanding Strict SPF (SR-Algo 1) and IGP Shortcuts

IN THIS SECTION

Strict SPF (SR-Algo 1) and IGP shortcut provides the following benefits

Benefits of Strict SPF (SR-Algo 1) and IGP Shortcuts

- Enhances segment routing capabilities.
- Helps to avoid loops by creating SR-TE tunnel to forward the traffic using the shortest IGP path.
- Ability to use SR-Algo 1 (strict SPF) along with SR-Algo 0 (default SPF) by default, when you enable SPRING.

Overview of Strict SPF (SR-Algo 1) and IGP Shortcuts

Segment routing (SR) simplifies operations and reduces resource requirements in the network by removing network state information from intermediate routers and placing path information into packet headers at the ingress node. However, in some cases, when there are nested SR-TE tunnels present and devices forward traffic over these SR-TE tunnel, traffic might loop, cause congestion, and not forward traffic over the shortest IGP path.

You can advertise SR algorithm 1 (strict SPF) and use the strict SPF SIDs to create SR-TE tunnels. Such SR-TE tunnels use only the strict path SPF instead of the local policy to reach the tunnel endpoint. You can specify prefixes in the import policy, based on which the tunnels redirect the traffic to a certain destination. Additionally, you can use SR-Algo 1 (strict SPF) along with SR-Algo 0 (default SPF) by default when you enable SPRING.

You can advertise strict-SPF SIDs in IS-IS LSPDU and use these SIDs to create SR-TE tunnel to forward the traffic through the shortest IGP path while not causing loops. Labeled IS-IS routes will then use the tunnel with the pre-defined shortcut statement at the `inet-mpls family` or `inet6-mpls family` configuration when you prefer `spring-te tunnel`.

The following illustration depicts the difference between SR-TE tunnels created without strict SPF SIDs and SR-TE tunnels created by using strict SPF (SR-Algo 1) SIDs:

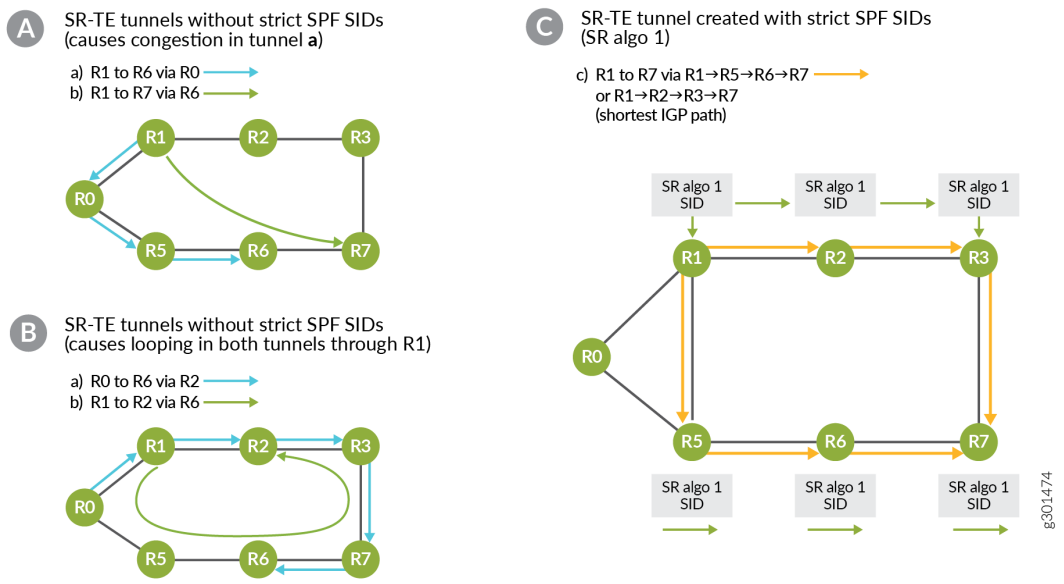


Figure A shows a network topology where SR-TE tunnel is not created using shortest IGP path to forward a traffic when a pre-existing SR-TE tunnel (or RSVP tunnel) is selected as ingress at R1. Two SR-TE tunnels exist in this topology. One from R1 to R6 (tunnel a, blue colored) via R0 and another tunnel is R1 to R7 (tunnel b, green colored) via R6. In this case, tunnel (b) is not created using the shortest IGP path. Thus, instead of taking the existing tunnel to reach R6 and then forwarding to R7, since, `inet-mpls` shortcut statement is enabled on R1, label IS-IS route uses the SR-TE tunnel (a) to forward the traffic destined to R7 avoiding the shortest IGP path, resulting traffic congestion on tunnel (a).

Figure B shows a topology where traffic loops. When the labeled IS-IS route chooses SR-TE tunnel as ingress and redirected to another SR-TE tunnel, then traffic will loop. In this topology we have two SR-TE tunnels, one from R0 to R6 via R2 and another tunnel is from R1 to R2 via R6. For a packet sent from R0 to R6 node, at R0 if this node picks SR-TE tunnel as ingress for the destination 2.2.2.6, it will push R2 label and forward to R1. At R1, another SR-TE tunnel is present via R6 with a label in mpls.0 table. When R1 receives this traffic to reach R2, it will use L-ISIS route shortcut over SR-TE tunnel and push R6 with the same label then forward to R0 node. At R0, the top label is the same as R6, which means that if the SR-TE tunnel again then it will push R2 label and forward the traffic to R1, which will loop.

Figure C shows the SR-TE tunnels created using Strict SPF SIDs that now supports SR-Algo 1 along with the pre-existing SR-Algo 0. Strict-SPF SID routes are installed in IS-IS only if the next-hop node is also capable of SR algo 1. Else, the traffic will be dropped. If you created the SR-TE tunnel using strict SPF SIDs and if anywhere on the path where a device did not advertise support for SR Algo 1, the tunnel will stay down. When tunnel is created using Strict SPF SIDs it will take the shortest IGP path to reach another tunnel endpoint, and thereby, avoids congestion. In a scenario where traffic loops (as shown in figure 2), the strict-SPF SIDs will be advertised in IS-IS LSPDU only by each node that is participating in SR domain that supports SR Algo 1. There can be multiple SR-TE tunnels, either created by using Strict-SPF SIDS or normal SIDs. When the operator configures the statement “use-for-shortcut” before creating the explicit route object (ERO), tunnels are created using strict SPF SIDs.

NOTE: This topic provides only a configuration overview of SR-TE needed for IGP shortcuts.

What's Next?

Stay tuned for more information about the working principles of SR-TE!

Example: Configure Strict SPF SIDs and Enable IGP Shortcuts in SPRING for IS-IS Protocol

IN THIS SECTION

Overview

Typically, when there are nested SR-TE tunnels present in a network and devices forward traffic over these SR-TE tunnels, traffic might not get forwarded over the shortest IGP path. As a result, traffic might loop.

You can advertise SR algorithm and use the strict SPF SIDs to create SR-TE tunnels to forward the traffic using shortest IGP path to avoid loop. Labeled IS-IS route will now use this tunnel with the pre-defined shortcut knob present under `inet-mpls` family (or `inet6-mpls` family) when you enable `spring-te`.

Requirements

IN THIS SECTION

This example uses the following hardware and software components:

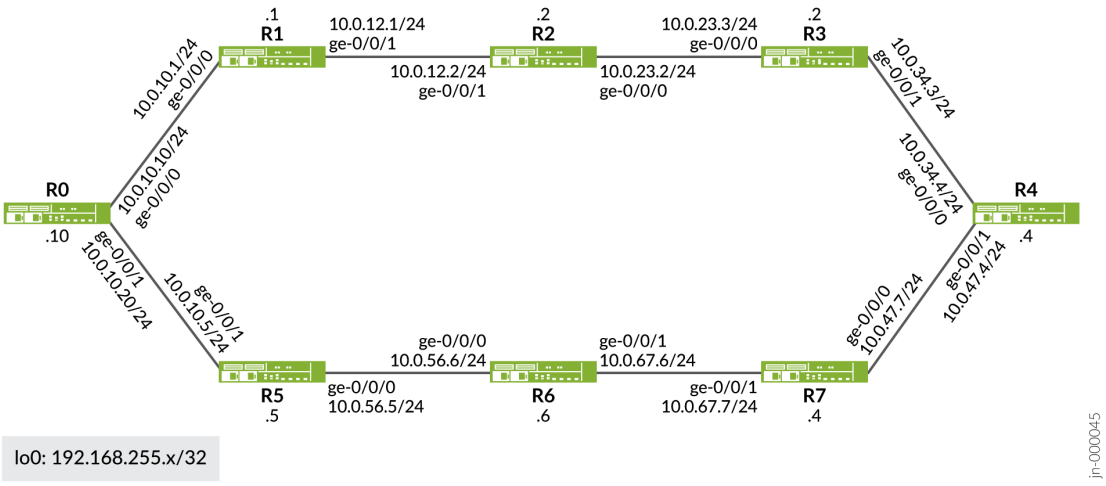
- Eight MX Series routers
- Junos OS Release 21.1R1 or later running on all devices

Topology

In the following example, we are demonstrating how loops occur in a network with multiple SR-TE tunnels and how you can resolve it by using strict SPF SIDs created by SR Algorithm 1. The example topology has two SR-TE tunnels. Tunnel A from R0>R2>R6 and Tunnel B from R1>R6>R2.

On R0 a packet destined to R6 typically use the IGP shortest path: that is, R0>R5>R6. When you configure an SR-TE tunnel with its ingress node as R0 (tunnel A), the packet needs to go through R2 as its first hop (destination: R6 and label: 403002), which means the traffic destined to R6 needs to take the R0>R2>R6 path. To reach R2, the packet needs to reach R1 first on the R0—R1 interface with the first label 403002. The R2's label 403002 should get forwarded from R1>R2 with no changes to the label stack. However, there is a second SR-TE tunnel (tunnel B) configured on R1 (R1>R6>R2) with destination R2 and label 403006. The packet that came from R0 with top label as R2 (403002) on R1 ends up using the second tunnel to reach R6. But to reach R6 on R1, R1—R0 (R1>R0>R5>R6) is the interface it needs to use. Thus, the packet reaches R0 again and the whole process repeats, resulting in looping.

With the SR algorithm 1 activated on all devices, and its labels activated on the relevant devices, when the packet from the ingress device R0 to the destination device R6 reaches R1 (tunnel A), the packet gets forwarded to R2. Even though R1 has LSP configured to consider R6 as its next hop (tunnel B), it would instead take the IGP shortest path (R1>R2). From R2, it reaches R6 through Tunnel A.



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: Depending on the type of MPC in your MX Series routers you might need to explicitly enable enhanced IP services to support the IS-IS delay feature. When you commit the `set chassis network-services enhanced-ip` configuration statement, you will be prompted to reboot the system.

R0

```
set system host-name R0
set system ports console log-out-on-disconnect
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 10.0.10.10/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.10.20/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.10/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5010.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.10/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3100
set policy-options policy-statement sspf term 1 then prefix-segment index 3000
```

```
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
set protocols isis interface ge-0/0/0.0
set protocols isis interface ge-0/0/1.0
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1000
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering tunnel-source-protocol spring-te
set protocols isis traffic-engineering family inet-mpls shortcuts
set protocols isis export sspf
set protocols mpls traceoptions file sspf-igp-short
set protocols mpls traceoptions file size 100m
set protocols mpls traceoptions file world-readable
set protocols mpls traceoptions flag ted-export
set protocols mpls traceoptions flag ted-import
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
deactivate protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
deactivate protocols rsvp interface fxp0.0 disable
set protocols source-packet-routing segment-list v4R0R7 h1 label 403102
set protocols source-packet-routing source-routing-path V4_R7 use-for-shortcut
set protocols source-packet-routing source-routing-path V4_R7 to 192.168.255.6
set protocols source-packet-routing source-routing-path V4_R7 primary v4R0R7
set routing-options router-id 192.168.255.10
set routing-options autonomous-system 65540
```

R1

```
set system host-name R1
set system ports console log-out-on-disconnect
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 10.0.10.1/24
set interfaces ge-0/0/0 unit 0 family iso
```

```
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.12.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.1/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5001.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.1/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3101
set policy-options policy-statement sspf term 1 then prefix-segment index 3001
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1001
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering tunnel-source-protocol spring-te
set protocols isis traffic-engineering family inet-mpls shortcuts
set protocols isis export sspf
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols source-packet-routing segment-list v4R1R2 h1 label 403106
set protocols source-packet-routing source-routing-path V4_R2 use-for-shortcut
set protocols source-packet-routing source-routing-path V4_R2 to 192.168.255.2
set protocols source-packet-routing source-routing-path V4_R2 primary v4R1R2
set routing-options router-id 192.168.255.1
set routing-options autonomous-system 65540
```

R2

```
set system host-name R2
set system ports console log-out-on-disconnect
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 10.0.23.2/24
```

```
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.12.2/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.2/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5002.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.2/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3102
set policy-options policy-statement sspf term 1 then prefix-segment index 3002
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1002
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis level 2 wide-metrics-only
set protocols isis level 1 wide-metrics-only
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering tunnel-source-protocol spring-te
set protocols isis traffic-engineering family inet-mpls shortcuts
set protocols isis export sspf
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols source-packet-routing segment-list v4R2R6 h1 label 403100
set protocols source-packet-routing segment-list v4R2R6 h2 label 403107
set protocols source-packet-routing source-routing-path v4_R6 use-for-shortcut
set protocols source-packet-routing source-routing-path v4_R6 to 192.168.255.2
set protocols source-packet-routing source-routing-path v4_R6 primary v4R2R6
set routing-options router-id 192.168.255.2
set routing-options autonomous-system 65540
```

R3

```
set system host-name R3
set system ports console log-out-on-disconnect
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 10.0.23.3/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.34.3/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.3/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5003.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.3/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3103
set policy-options policy-statement sspf term 1 then prefix-segment index 3003
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1003
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis level 2 wide-metrics-only
set protocols isis level 1 wide-metrics-only
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis export sspf
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols source-packet-routing
set routing-options router-id 192.168.255.3
set routing-options autonomous-system 65540
```

R4

```
set system host-name R4
set system ports console log-out-on-disconnect
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 10.0.34.4/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.47.4/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.4/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5004.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.4/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3104
set policy-options policy-statement sspf term 1 then prefix-segment index 3004
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1004
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis level 2 wide-metrics-only
set protocols isis level 1 wide-metrics-only
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis export sspf
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols source-packet-routing
set routing-options router-id 192.168.255.4
set routing-options autonomous-system 65540
```

R5

```
set system host-name R5
set system ports console log-out-on-disconnect
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 10.0.56.5/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.10.5/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.5/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5005.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.5/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3105
set policy-options policy-statement sspf term 1 then prefix-segment index 3005
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1005
set protocols isis source-packet-routing node-segment ipv6-index 2005
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis level 2 wide-metrics-only
set protocols isis level 1 wide-metrics-only
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis export sspf
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols source-packet-routing
set routing-options router-id 192.168.255.5
set routing-options autonomous-system 65540
```

R6

```
set system host-name R6
set system ports console log-out-on-disconnect
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 10.0.56.6/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.67.6/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.6/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5006.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.6/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3106
set policy-options policy-statement sspf term 1 then prefix-segment index 3006
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1006
set protocols isis source-packet-routing node-segment ipv6-index 2006
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis level 2 wide-metrics-only
set protocols isis level 1 wide-metrics-only
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis export sspf
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols source-packet-routing
set routing-options router-id 192.168.255.6
set routing-options autonomous-system 65540
```

R7

```
set system host-name R7
set system ports console log-out-on-disconnect
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 10.0.47.7/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.67.7/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.7/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5007.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.7/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3107
set policy-options policy-statement sspf term 1 then prefix-segment index 3007
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1007
set protocols isis source-packet-routing node-segment ipv6-index 2007
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis level 2 wide-metrics-only
set protocols isis level 1 wide-metrics-only
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis export sspf
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols source-packet-routing
set routing-options router-id 192.168.255.7
set routing-options autonomous-system 65540
```

Enable Default SIDs (Algorithm 0) in SPRING

1. Configure the basic device settings such as hostname, IPv4 address, loopback interface address, NET address, family ISO, family MPLS (with maximum number of labels for segment routing routed paths), `enhanced-ip` mode, router-ID, and autonomous system (AS) number on all eight routers.

```
user@R0#
set chassis network-services enhanced-ip
set system host-name R0
set interfaces ge-0/0/0 unit 0 family inet address 10.0.10.10/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 8
set interfaces ge-0/0/1 unit 0 family inet address 10.0.10.20/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 8
set interfaces lo0 unit 0 family inet address 192.168.255.10/32
set interfaces lo0 unit 0 family iso address 49.1921.6825.5010.00
set interfaces lo0 unit 0 family mpls maximum-labels 8
set routing-options router-id 192.168.255.10
set routing-options autonomous-system 65540
```

2. Enable IS-IS, RSVP, and MPLS protocols on all interfaces of all eight devices. You can also specify trace files and operations for MPLS.

```
user@R0#
set protocols isis interface ge-0/0/0.0
set protocols isis interface ge-0/0/1.0
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols mpls traceoptions file sspf-igp-short
set protocols mpls traceoptions file size 100m
set protocols mpls traceoptions file world-readable
set protocols mpls traceoptions flag ted-export
set protocols mpls traceoptions flag ted-import
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols rsvp interface all
deactivate protocols rsvp interface all
```

```
set protocols rsvp interface fxp0.0 disable
deactivate protocols rsvp interface fxp0.0 disable
```

3. Configure all routers to advertise their loopback address and specify the index and the node segment of the prefix segment.

```
user@R0#
set policy-options policy-statement sspf term 1 from route-filter 192.168.255.10/32 exact
set policy-options policy-statement sspf term 1 then prefix-segment index 3000
set policy-options policy-statement sspf term 1 then prefix-segment node-segment
set policy-options policy-statement sspf term 1 then accept
```

4. Configure the start-label and index-range of SRGB for SPRING. Configure the value of IPv4 node segment index and assign 128 flex algorithm.

```
user@R0#
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 64000
set protocols isis source-packet-routing node-segment ipv4-index 1000
set protocols isis source-packet-routing flex-algorithm 128
```

5. Configure options for shortest-path-first (SPF) algorithm in IS-IS protocol to enable the source packet routing node segment labels for computing backup paths on R0, R1, and R2. Set maximum labels set to 8.

```
user@R0#
set protocols isis backup-spf-options use-post-convergence-lfa maximum-labels 8
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
```

6. Configure traffic engineering options to choose label switched paths from `spring-te` and use the MPLS paths as next hops on R0, R1, and R2. Set the IS-IS export policy.

```
user@R0#
set protocols isis traffic-engineering tunnel-source-protocol spring-te
set protocols isis traffic-engineering family inet-mpls shortcuts
set protocols isis export sspf
```

7. Configure an R2 label 403002 (created for algorithm 0 to create default SPF SID) on R0 with R2 as its next hop to the destination R6 and enable `use-for-shortcut`. Create labels on R1, and R2 as well.

```
user@R0#
set protocols source-packet-routing segment-list v4R0R7 h1 label 403002
set protocols source-packet-routing source-routing-path V4_R7 use-for-shortcut
set protocols source-packet-routing source-routing-path V4_R7 to 192.168.255.6
set protocols source-packet-routing source-routing-path V4_R7 primary v4R0R7
```

```
user@R1#
set protocols source-packet-routing segment-list v4R1R2 h1 label 403006
set protocols source-packet-routing source-routing-path V4_R2 use-for-shortcut
set protocols source-packet-routing source-routing-path V4_R2 to 192.168.255.2
set protocols source-packet-routing source-routing-path V4_R2 primary v4R1R2
```

```
user@R2#
set protocols source-packet-routing segment-list v4R2R6 h1 label 403000
set protocols source-packet-routing segment-list v4R2R6 h2 label 403007
set protocols source-packet-routing source-routing-path v4_R6 use-for-shortcut
set protocols source-packet-routing source-routing-path v4_R6 to 192.168.255.2
set protocols source-packet-routing source-routing-path v4_R6 primary v4R2R6
```

8. Enter `commit` command to commit the configurations.

Enable Strict SPF SIDs (Algorithm 1) in SPRING

1. To replace the labels used for default SPF SIDs with labels to be used for strict SPF SIDs, configure the following:

```
user@R0#
delete protocols source-packet-routing segment-list v4R0R7 h1 label 403002
set protocols source-packet-routing segment-list v4R0R7 h1 label 403102
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3100
```

```
user@R1#
delete protocols source-packet-routing segment-list v4R1R2 h1 label 403006
```

```
set protocols source-packet-routing segment-list v4R1R2 h1 label 403106
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3101
```

```
user@R2#
delete protocols source-packet-routing segment-list v4R2R6 h1 label 403000
set protocols source-packet-routing segment-list v4R2R6 h1 label 403100
delete protocols source-packet-routing segment-list v4R2R6 h2 label 403007
set protocols source-packet-routing segment-list v4R2R6 h2 label 403107
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3102
```

2. Set/activate algorithm 1 on all other routers in the network.

```
user@R3#
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3103
```

```
user@R4#
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3104
```

```
user@R5#
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3105
```

```
user@R6#
 set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3106
```

```
user@R7#
set policy-options policy-statement sspf term 1 then prefix-segment algorithm 1 index 3107
```

3. Enter `commit` command to commit all configuration.

Results

Check the results of the configuration:

```
user@R0# show
system {
    host-name R0;
    ports {
        console log-out-on-disconnect;
    }
}
chassis {
    network-services enhanced-ip;
}
interfaces {
    ge-0/0/0 {
        unit 0 {
            family inet {
                address 10.0.10.10/24;
            }
            family iso;
            family mpls {
                maximum-labels 8;
            }
        }
    }
    ge-0/0/1 {
        unit 0 {
            family inet {
                address 10.0.10.20/24;
            }
            family iso;
            family mpls {
                maximum-labels 8;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.255.10/32;
            }
```

```
            family iso {
                address 49.1921.6825.5010.00;
            }
            family mpls {
                maximum-labels 8;
            }
        }
    }
}
policy-options {
    policy-statement sspf {
        term 1 {
            from {
                route-filter 192.168.255.10/32 exact;
            }
            then {
                prefix-segment {
                    algorithm 1 index 3100;
                    index 3000;
                    node-segment;
                }
                accept;
            }
        }
    }
}
protocols {
    isis {
        interface ge-0/0/0.0;
        interface ge-0/0/1.0;
        interface all;
        interface fxp0.0 {
            disable;
        }
        interface lo0.0 {
            passive;
        }
        source-packet-routing {
            srgb start-label 400000 index-range 64000;
            node-segment ipv4-index 1000;
            flex-algorithm 128;
        }
        backup-spf-options {
```

```
            use-post-convergence-lfa {
                maximum-labels 8;
                maximum-backup-paths 8;
            }
            use-source-packet-routing;
        }
        traffic-engineering {
            tunnel-source-protocol {
                spring-te;
            }
            family inet-mpls {
                shortcuts;
            }
        }
        export sspf;
    }
    mpls {
        traceoptions {
            file sspf-igp-short size 100m world-readable;
            flag ted-export;
            flag ted-import;
        }
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    rsvp {
        inactive: interface all;
        interface fxp0.0 {
            inactive: disable;
        }
    }
    source-packet-routing {
        segment-list v4R0R7 {
            h1 label 403102;
        }
        source-routing-path V4_R7 {
            use-for-shortcut;
            to 192.168.255.6;
            primary {
                v4R0R7;
            }
```

```
        }
    }
}
routing-options {
    router-id 192.168.255.10;
```

Verification

IN THIS SECTION

Verify IS-IS Adjacencies

IN THIS SECTION

Purpose

Verify expected IS-IS adjacencies on the routing devices.

Action

From operational mode, enter the `show isis adjacency` command.

```
user@R0> show isis adjacency
Interface             System         L State       Hold (secs) SNPA
ge-0/0/0.0            R1             1  Up                    23  56:4:15:0:1c:d2
```

```
ge-0/0/0.0          R1          2  Up                  25  56:4:15:0:1c:d2
ge-0/0/1.0          R5          1  Up                  25  56:4:15:0:1c:eb
ge-0/0/1.0          R5          2  Up                  24  56:4:15:0:1c:eb
```

Meaning

The output indicates that R0 has successfully formed IS-IS adjacencies on its `ge-0/0/0.0` and `ge-0/0/1.0` interfaces, which attach to their R1 and R5 routers, respectively.

Verify Route Table inet.3

IN THIS SECTION

Purpose

Verify the `inet.3` routing table with the advertised.

Action

From operational mode, enter the `show route table inet.3` command.

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

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

192.168.255.1/32   *[L-ISIS/14] 3d 19:43:17, metric 10
                    >  to 10.0.10.1 via ge-0/0/0.0
192.168.255.2/32   *[L-ISIS/14] 3d 19:43:17, metric 20
                    >  to 10.0.10.1 via ge-0/0/0.0, Push 403002
192.168.255.3/32   *[L-ISIS/14] 3d 19:43:17, metric 30
                    >  to 10.0.10.1 via ge-0/0/0.0, Push 403003
192.168.255.4/32   *[L-ISIS/14] 3d 19:43:17, metric 21
                    >  to 10.0.10.1 via ge-0/0/0.0, Push 403004, Push 403002(top)
192.168.255.5/32   *[L-ISIS/14] 3d 19:43:17, metric 10
```

```
                    >  to 10.0.10.5 via ge-0/0/1.0
192.168.255.6/32   *[SPRING-TE/8] 3d 19:43:17, metric 1, metric2 20
                    >  to 10.0.10.1 via ge-0/0/0.0, Push 403002
                    [L-ISIS/14] 3d 19:43:17, metric 1
                    >  to 10.0.10.1 via ge-0/0/0.0, Push 403002
192.168.255.7/32   *[L-ISIS/14] 3d 19:43:17, metric 11
                    >  to 10.0.10.1 via ge-0/0/0.0, Push 403007, Push 403002(top)
```

Meaning

Th output displays the routes on inet.3 table.

Verify Route Label (Default SPF)

IN THIS SECTION

Purpose

Verify route labels created for default SPF on the routing devices.

Action

From operational mode, enter the `show route label 403002` command.

```
user@R0> show route label 403002

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

403002             *[L-ISIS/14] 3d 20:17:24, metric 20
                    >  to 10.0.10.1 via ge-0/0/0.0, Swap 403002
```

```
regress@R1> show route label 403002
```

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

403002             *[L-ISIS/14] 3d 20:31:53, metric 1
                    >  to 10.0.10.10 via ge-0/0/0.0, Push 403006
403002(S=0)        *[L-ISIS/14] 3d 20:31:53, metric 1
                    >  to 10.0.10.10 via ge-0/0/0.0, Push 403006
```

Meaning

The output indicates that the packet is pushing R2's label 403002 to R1 to reach its next hop R2. But on R1, it picks up the tunnel B and pushes the label of its next hop R6- 403006, instead of getting forwarded from R1 to R2 on tunnel A.

Verify Route Label (Strict SPF)

IN THIS SECTION

Purpose

Verify route labels created for strict SPF on the routing devices.

Action

From operational mode, enter the `show route label 403102` command.

```
user@R0> show route label 403102

mpls.0: 32 destinations, 32 routes (32 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
403102             *[L-ISIS/14] 00:36:07, metric 20
                    >  to 10.0.10.1 via ge-0/0/0.0, Swap 403102
```

```
regress@R1> show route label 403102

mpls.0: 32 destinations, 32 routes (32 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

403102             *[L-ISIS/14] 00:37:38, metric 10
                    >  to 10.0.12.2 via ge-0/0/1.0, Pop
403102(S=0)        *[L-ISIS/14] 00:37:38, metric 10
                    >  to 10.0.12.2 via ge-0/0/1.0, Pop
```

Meaning

The first output indicates that the packet with R2's label has reached R1. The second output indicates that the packet is now forwarded to R2 (on tunnel A), instead of getting picked by the tunnel B on R1. Once it reaches R2, it can complete the tunnel A path and reach R6.

Overview of Segment Routing over RSVP Forwarding Adjacency in IS-IS

Segment routing architecture enables the ingress nodes in a core network to steer traffic through explicit paths through the network. The architecture provides the mechanism to enable source routing. Paths are encoded as sequences of topological subpaths called segments, which are advertised by link-state routing protocols such as IS-IS and OSPF.

A *forwarding adjacency* is a traffic engineered label-switched path (LSP) that is configured between two nodes and that is used by the interior gateway protocol (IGP) to forward traffic. The forwarding adjacency creates a tunneled path for sending data between peer devices in an RSVP LSP network.

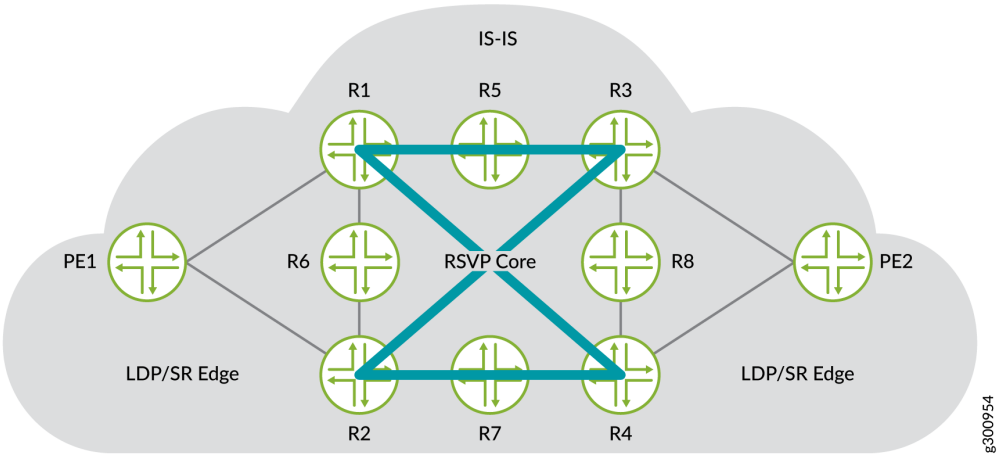
Junos OS supports segment routing traffic to be carried over RSVP LSPs that are advertised as forwarding adjacencies in IS-IS.

Benefits of Segment Routing over RSVP LSPs

- Reduces network complexity by removing protocols such as LDP.
- Leverages IGPs such as IS-IS, and RSVP for efficient and flexible forwarding.
- Provides a faster and more efficient way of forwarding traffic in the RSVP core network.

Figure 16 on page 169 illustrates the typical deployment network for segment routing over RSVP forwarding adjacency.

Figure 16: Segment Routing over RSVP Forwarding Adjacency



The network consists of provider edge (PE) routers configured with LDP on the edge and RSVP in the core. You can easily replace LDP with IS-IS segment routing because segment routing eliminates the need for MPLS signaling protocols such as LDP. As a result, you enable network simplification by removing a protocol from the network.

How IS-IS Segment Routing over RSVP Forwarding Adjacency Works

RSVP LSPs are configured as links in IS-IS. IS-IS builds dummy adjacencies over these links (no hellos) and advertises them as links in LSPs. Because RSVP LSPs are advertised as forwarding adjacencies, the LDP or segment routing edge nodes can forward traffic towards appropriate core nodes. The metric on RSVP LSPs is manipulated to manage traffic steering from the head node to the end nodes. RSVP uses the shortest-path-first (SPF) algorithm to compute the shortest path to all nodes in the network. As a result, when IP routes point to RSVP LSPs, segment routing routes also point to these LSPs. This is because segment routing reuses the SPF computation performed for the IP routes.

RELATED DOCUMENTATION

- *Understanding Source Packet Routing in Networking (SPRING)*

Traffic Engineering in IGPs with Segment Routing

IN THIS SECTION

Flexible Algorithms in IGP for Segment Routing

SUMMARY

A flexible algorithm allows IGPs alone to compute constraint based paths over the network thereby providing simple traffic engineering without using a network controller. This is a light weight solution for networks that have not implemented a controller with full fledged segment routing but still want to reap the benefits of segment routing in their network.

IN THIS SECTION

Understanding IGP Flexible Algorithms for Segment Routing

IN THIS SECTION

Define flexible algorithms that compute paths using different parameters and link constraints to thin slice a network based on your requirements. For example, you can define a flexible algorithm that computes a path to minimize IGP metric and define another flexible algorithm to compute a path based on traffic engineering metric to divide the network into separate planes. This feature allows networks without a controller to configure traffic engineering using segment routing without actually implementing a network controller. You can use the prefix SIDs to steer packets along the constraint-based paths. You can configure the prefix SIDs for flexible algorithm through policy configurations.

IGP protocols use a link metric to calculate a best path. However, the best IGP path might not always be the best path for certain types of traffic. Therefore, the IGP computed best path based on the shortest IGP metric is often replaced with traffic engineered path due to the traffic requirements that are not reflected by the IGP metric. Typically RSVP-TE or SR TE is used for computing the path based on additional metrics and constraints to overcome this limitation. Junos installs such paths in the forwarding tables in addition to or as a replacement for the original path computed by the IGPs.

Benefits of Configuring Flexible Algorithm

- A lightweight version of segment routing traffic engineering that can be used in the core of the network.
- Allows you to configure traffic engineering using segment routing even without installing a network controller.
- Compute TI-LFA backup path using the same flexible algorithm definition and constraints computation.
- Ability to provision constrained primary path based on a single label.

What is Flexible Algorithm Definition (FAD)?

A flexible algorithm allows IGP to calculate additional best paths based on specified constraints thereby providing simple traffic engineering without using a network controller. This is a lightweight solution for

networks that have not implemented a controller with full fledged segment routing but still want to reap the benefits of segment routing in their network. Every operator can define separate constraints or colors depending on their requirements.

To define a flexible algorithm, include `flex-algorithm` *id* statement at the `[edit routing-options]` hierarchy level. The flexible algorithm definition (FAD) is assigned with an identifier ranging from 128 through 255. This flexible algorithm can be defined on one or more routers in a network. A flexible algorithm computes a best path based on the following parameters:

- **Calculation type**—SPF or strict SPF are the two available calculation type options. You can specify one of these calculation types in your FAD. Select the SPF calculation type if you want to influence the SPF computation on your device based on a certain local policy such as traffic engineering shortcuts. If you select strict SPF then the local policy cannot influence the SPF path selection.

- **Metric type**- IGP metric, TE metric, or delay metric are the available metric type options. You can specify one of these metric types in your FAD depending on your network requirement. If you do not want to use the IGP metric for a specific link you can configure a TE metric that IS-IS can use for calculating the route.

- **Priority**- You can assign a priority to your FADs as per your requirement and IS-IS prioritizes a particular FAD advertisement over another FAD based on your assigned priority.

  > **NOTE**: For FADs with link-constraints to work, all relevant links should advertise the admin-colors in IS-IS, which means either RSVP is enabled on the interfaces or `set protocols isis traffic-engineering advertise always` is configured.

- **Set of Link constraints**- You can configure admin-groups for many protocols at the `[edit protocols mpls admin-groups]` hierarchy level to color an individual link. These `admin-groups` can then be defined as `include any`, `include-all` or `exclude` at the `[edit routing-options flex-algorithm` *definition* `admin-groups]` hierarchy level.

We recommend configuring flexible algorithm definitions on only a few routers to provide redundancy and to avoid conflicts. Flexible algorithm definition is advertised in IGP as `FAD sub-TLVs`. In very large networks, we do not recommend configuring more than 8 flexible algorithms as each flexible algorithm will compute its own path and might cause performance issues beyond that.

Its also recommended that you configure multiple FAD servers in a specific ISIS Level before configuring any devices to participate in that FAD. In the case of an ISIS L1/L2 node (ABR), it's also recommended that you configure the FAD at both ISIS Level 1 and Level 2. If a FAD is configured only on a single ABR, traffic drops over flex algorithm paths are possible if the routing process restarts on that ABR. Its therefore a good design practice to have multiple ABRs, each of which has the FAD configured at both ISIS levels.

The default FAD has the following parameters:

- calculation type: spf
- metric type: igp-metric
- priority: 0
- Link constraints: none

NOTE: Modifying the flexible algorithm definition in a live network or on the fly could cause traffic disruptions until all the nodes converge on the new paths.

We support flexible Algorithm Definition (FAD)" and "Flexible Algorithm Prefix Metric (FAPM)" in TED and implements two new corresponding TLVs "FAD TLV" and "FAPM TLV" in BGP-LS. The value of FAD TLV contains Flex-Algorithm, Metric-Type, Calculation-Type and Priority, all of which are one byte each. The TLV might have zero or more sub-TLVs included in it. The five sub-tlvs are Flex Algo Exclude Any Affinity, Flex Algo Include Any Affinity, Flex Algo Include All Affinity, Flex Algo Definition Flags and Flex Algo Exclude SRLG.

The FAD TLV can only be added to the BGP-LS Attribute of the Node NLRI if the corresponding node originates in the underlying IGP TLV or sub-TLV. The BGP-LS Attribute associated with a Node NLRI might include one or more FAD TLVs corresponding to the Flexible Algorithm Definition for each algorithm that the node is advertising.

The value of FAPM TLV contains Flex-Algorithm (1 byte), Reserved (3 bytes) and Metric (4 bytes). The FAPM TLV can be added to the BGP-LS Attribute of the Prefix NLRI originated by a node, only if the corresponding node originates from the Prefix.

We've defined the Flexible Algorithm Prefix Metric (FAPM) to allow optimal end-to-end path for an interarea prefix. The area border router (ABR) must include the FAPM when advertising the prefix between areas that is reachable in that given Flexible Algorithm (flex algo). When a prefix is unreachable, the ABR must not include that prefix in that flex algo when advertising between areas. The defined FAPM provides inter-area support.

We support delay normalization and Flexible Algorithm Definition (FAD) defined constraints related to admin-groups and shared risk link group (SRLG) as defined in RFC 9350, IGP Flexible Algorithm.

During flexible algorithm computation, when the measured latency values are not equal and the difference is insignificant, IS-IS advertises this slightly higher latency value as a metric. IS-IS uses this normalized latency delay value instead of the measured delay value.

To configure flexible algorithm application specific SRLG values, include the application-specific statement at the `[edit protocols isis interface interface-name level level]` hierarchy level. To exclude SRLG constraint in an FAD, include the `exclude-srlg` statement at the `[edit routing-options flex-algorithm name definition]` hierarchy level.

You can control path selection by configuring the preference for OSPF Flexible Algorithm routes in `inetcolor.0` and `mpls.0` routing tables.

Configure `flex-algorithm-preference` statement at the `[edit protocols ospf]` hierarchy level to prioritize desired routes and improve traffic engineering across IP and MPLS domains.

Participation in a Flexible Algorithm

You can configure specific routers to participate in a particular flexible algorithm as per your requirement. Paths computed based on a flexible algorithm definition is used by various applications each potentially using its own specific data plane for forwarding the data over such paths. The participating device must explicitly advertise its participation in a particular flexible algorithm to every application in the segment routing flexible algorithm sub TLV for IS-IS. You can configure a node to participate in a certain flexible algorithm provided it can support the constraints specified in that FAD.

To configure participation in a flexible algorithm include the `flex-algorithm` statement at the `[edit protocols isis source-packet- routing]` hierarchy level. The same device can advertise a FAD and also participate in a flexible algorithm.

Network Topology Configured with Flexible Algorithm Definitions

Figure 17 on page 175 shows the sample topology, there are 8 routers R0, R1, R2, R3, R4, R5, R6, and R7. Four flexible algorithms, 128, 129, 130, and 135 are defined and configured with admin-groups as listed in the following table:

Table 1: Flex Algorithm Definition (FAD) – Color Rules

Flex Algorithm Definition (FAD)	Color
128	Include any Red
129	Include any Green
130	Include any Green and Blue
135	Exclude Red

Figure 17: Flexible Algorithm Topology

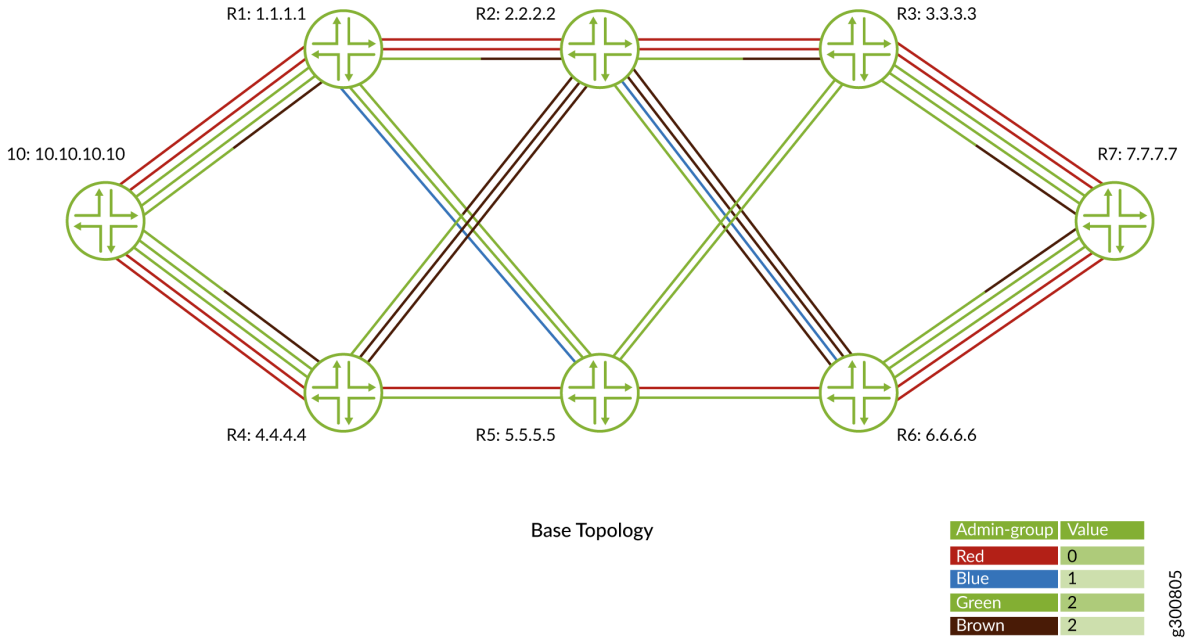


Figure 18 on page 176 shows how FAD 128 routes traffic on any interface that is configured with admin group red.

Figure 18: Traffic Flow for Flexible Algorithm Definition 128

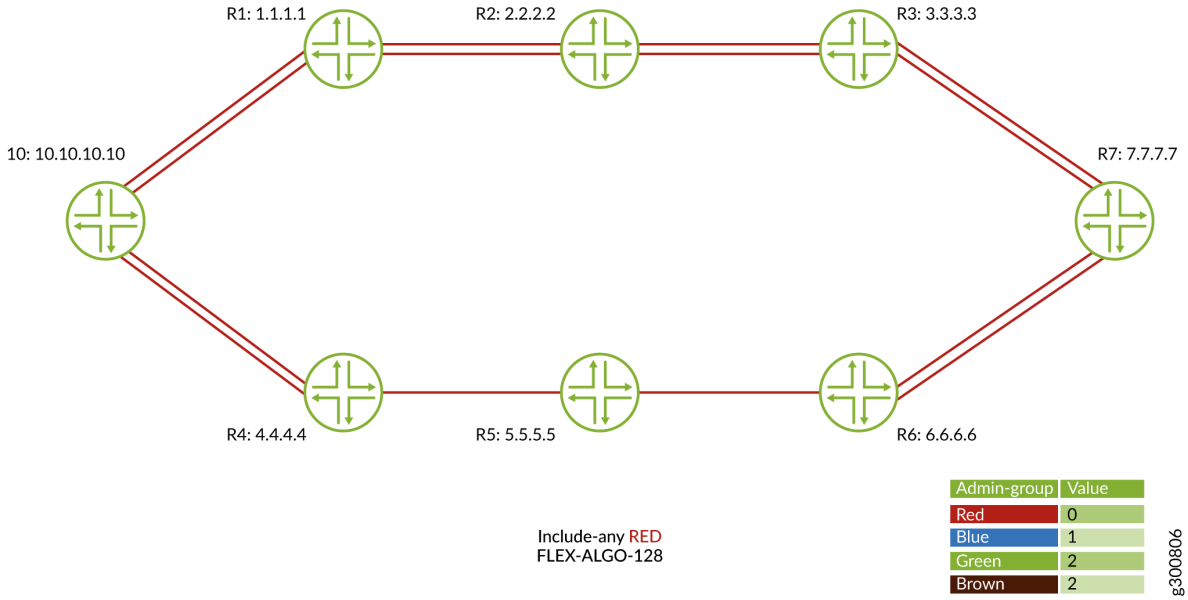


Figure 19 on page 177 shows how FAD 129 routes traffic on any interface that is configured with admin group green.

Figure 19: Traffic Flow for Flexible Algorithm Definition 129

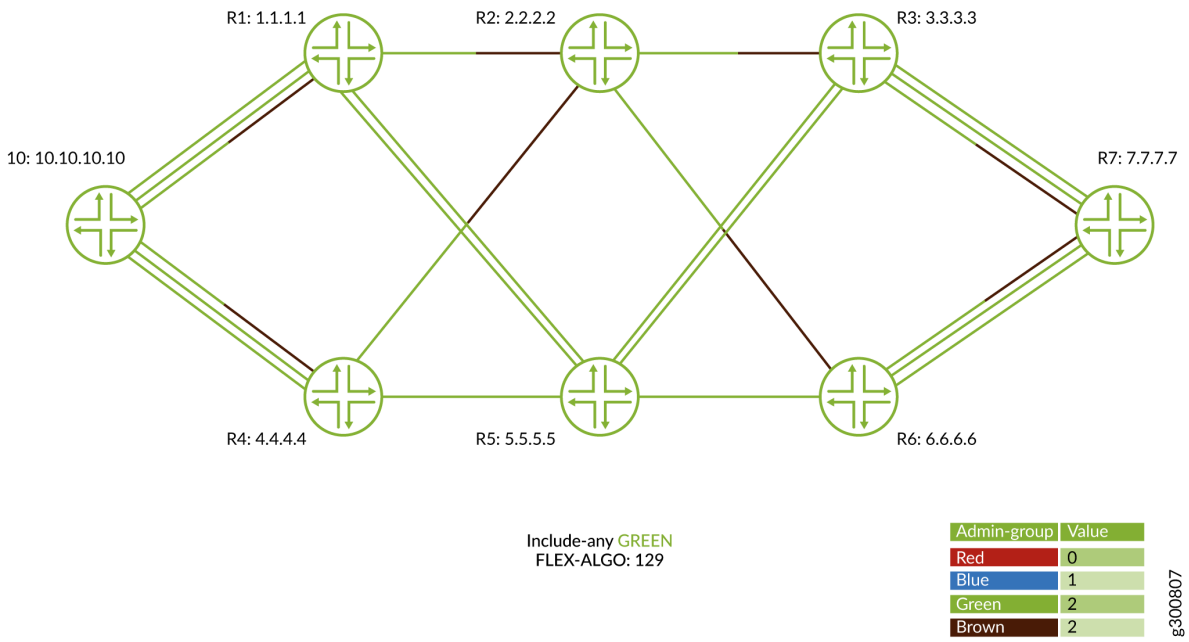


Figure 20 on page 178 shows how FAD 130 routes traffic on any interface that is configured with admin group green and blue.

Figure 20: Traffic flow for Flexible Algorithm Definition 130

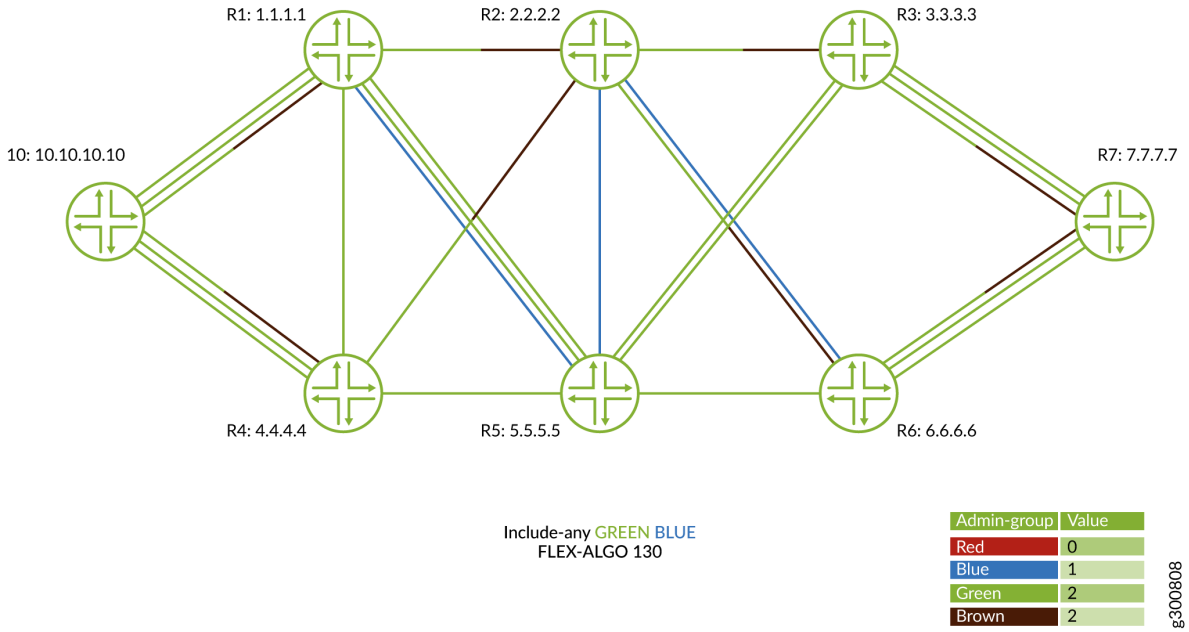
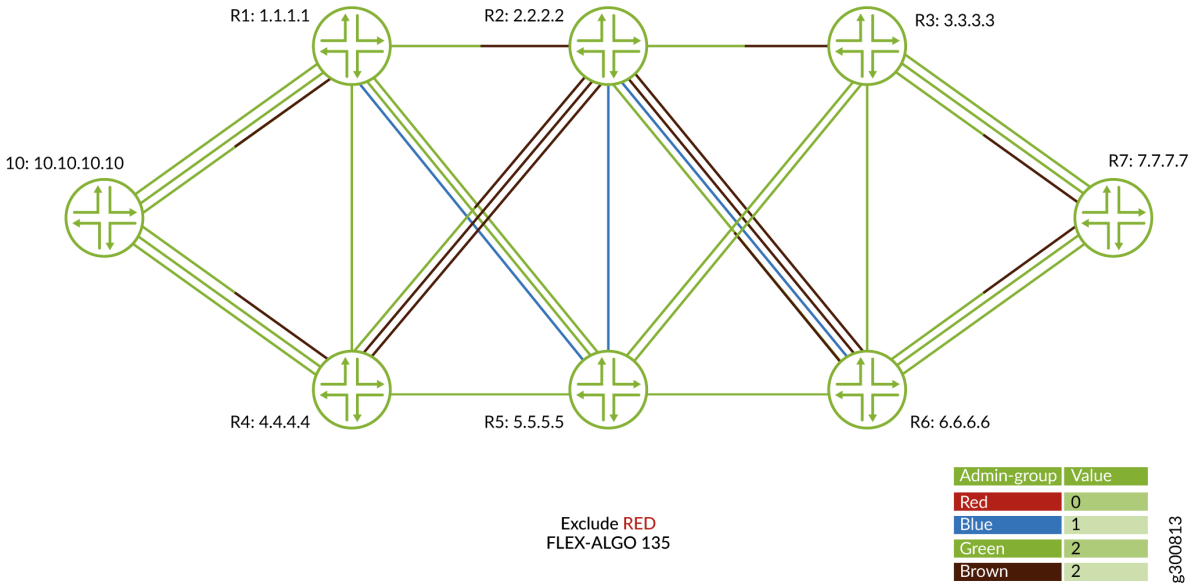


Figure 21 on page 179 shows how FAD 135 routes traffic on any interface that is not configured with admin group red.

Figure 21: Traffic Flow for Flexible Algorithm Definition 135



Flexible Algorithm RIBs

For every flexible algorithm that a router participates in, the corresponding flexible algorithm routes are installed in the corresponding flexible algorithm RIB groups also known as routing tables. By default, labeled flexible algorithm routes are installed in the inet.color, `inet(6)color.0` and `mpls.0` RIBs. They could also be installed in colored RIBs, such as `junos-rti-tc-<color>.inet(6).3` when `use-transport-class` statement is configured under `routing-options flex-algorithm <id>`. For more information, see "Flexible Algorithm and Flexible Algorithm Prefix Metrics Leaking across IS-IS Multi-Instance" on page 385.

BGP Community and Flexible Algorithms

A flexible algorithm can be associated with a color. When a service prefix, such as a VPN service carries a BGP color extended community, by default the BGP service prefix resolves a flex-algo route that has the same associated color value. The flexible algorithm ingress routes that are installed in the `inet(6)color.0` tables will have this color value associated with the route. However, you can configure a different associate color value at the `[edit routing-options flex-algorithm` *`id`* `color` *`color`*`]` hierarchy level.

NOTE: Changing the associated color value in a flexible algorithm might result in traffic disruption. If you modify the color in a flexible algorithm definition, all routes pertaining to that flexible algorithm are removed from the RIB and added again with the new color.

You can leak BGP-LU prefixes into the IGP with flexible algorithm prefix-SIDs. For more information, see "Leaking BGP-LU Prefixes into Flexible Algorithm" on page 387.

You can now leak BGP-CT prefixes into flexible algorithm and vice-versa. For more information, see "Leaking BGP-CT Prefixes into Flexible Algorithm" on page 388.

Application-specific Link Attribute based flexible algorithm

You can advertise different te-attributes such as te-metric, delay-metric, or admin-groups for RSVP and flexible algorithms on the same link. This is done using flexible algorithm specific application-specific link attribute as defined in RFC 8920.

The advantage of having a flexible algorithm application-specific link attribute advertise te-metric, delay-metric, or admin-groups is that a single link can advertise different te-link-attributes for legacy applications such as RSVP and different te-link-attributes for flexible algorithms.

To configure flexible algorithm application-specific te-attribute, include the `application-specific` statement at the `[edit protocols ospf area interface]` hierarchy level and the `strict-asla-based-flex-algorithm` statement at the `[edit protocols ospf source-packet-routing]` hierarchy level. With this implementation, it is no longer mandatory for the link to have RSVP enabled and `[edit protocols ospf traffic-engineering advertisement always]` to be configured which is the case with the existing behavior of advertising traffic engineering attributes.

NOTE: The Junos OS and Junos OS Evolved implementation of application-specific link attribute supports flexible algorithm applications only.

Strict Application-Specific Link Attribute based flexible algorithm

The default behavior of application-specific flexible algorithm is to use the flexible algorithm application-specific te-attributes for a link if available, and if not, then fall back to the common application-specific te-attributes, and if neither are available, use the legacy te-attributes.

The configuration statement `strict-asla-based-flex-algorithm` at the `[edit protocols ospf source-packet-routing]` has to be applied to all the flexible algorithms running on the devices in the network to avoid routing loops.

If `strict-asla-based-flex-algorithm` is configured on all the devices, either a common application-specific te-attribute or flexible algorithm application-specific te-attribute must be advertised for each flexible algorithm link. In the absence of application-specific te-attributes, the device does not fall back to the legacy te-attributes and simply ignores the link.

The Operating System supports the following features in conjunction with application-specific link attribute based flexible algorithm:

- The application-specific te-attribute subTLV to comply with RFC 8920. The application-specific te-attributes sub-TLV is a sub-TLV of the OSPFv2 extended link TLV as defined in RFC 7684.
- Partially supports standard application identifier bit mask to advertise X-bit for flexible algorithms. Only the te-metric, delay-metric, or admin-groups are advertised as part of the application-specific link attribute sub-TLV.

The Operating System does not support the following features in conjunction with application-specific link attribute based flexible algorithm:

- Advertising user-defined application identifier bit masks is not supported.
- Readvertising flexible algorithm application-specific link attribute or rather any application-specific link attributes with BGP-LS is not supported because Traffic Engineering Database (TED) does not support application-specific link attribute.
- Advertising a common application-specific link attribute with standard application identifier bit mask and user-defined application identifier bit masks length set to zero is not supported.
- Advertising SRLG link constraint in flexible algorithm is not supported.
- Supporting traffic engineering for multiple applications is not supported, except for flexible algorithms.
- Defining admin-groups independent of MPLS is not supported.

Supported and Unsupported Features

Junos OS supports flexible algorithms in the following scenarios:

- Support for configuring and advertising prefix SIDs for different flexible algorithms.
- Partially supports RFC 9350, *IGP Flexible Algorithm*
- Inter-level (IS_IS) leaking of flexible algorithm prefix SIDs is supported.
- The current implementation for flexible algorithms is supported for only OSPFv2 only as only OSPFv2 supports segment routing.

Junos OS does not support the following features in conjunction with flexible algorithms:

- Flexible algorithm is applicable only for default unicast topology, OSPFv2 multi-topology is not supported.
- IS-IS shortcuts and other IS-IS traffic engineering configuration options are not applicable for flexible algorithm computation
- Prefix and SID conflict resolution is not supported.

- Remote loop free alternate functionality is not supported because TI-LFA is the preferred FRR computation
- OSPFv2 shortcuts and other OSPFv2 traffic engineering configuration options are not applicable for flexible algorithm computation.
- Advertising flexible algorithm definition in the absence of flexible algorithm participation is not supported for OSPFv2.
- Extended Admin-Groups (EAG) are not supported because they are not supported in IS-IS.

SEE ALSO

Configuring Flexible Algorithm for Segment Routing | **183**

flex-algorithm

flex-algorithm

definition

show isis flex-algorithm

flex-algorithm (Protocols OSPF)

definition (Protocols OSPF)

application-specific (Protocols OSPF)

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
25.4R1	Starting in Junos OS and Junos OS Evolved Release 25.4R1, you can control path selection by configuring the preference for OSPF Flexible Algorithm routes in `inetcolor.0` and `mpls.0` routing tables.
22.4R1	Starting in Junos OS Release 22.4R1, we've defined the Flexible Algorithm Prefix Metric (FAPM) to allow optimal end-to-end path for an interarea prefix.
22.2R1	Starting in Junos OS and Junos OS Evolved Release 22.2R1, you can advertise different te-attributes such as te-metric, delay-metric, or admin-groups for RSVP and flexible algorithms on the same link.

Configuring Flexible Algorithm for Segment Routing

Before you begin configuring the flexible algorithm for IS-IS, make sure you:

1. Configure the device interfaces to enable IP transport.
2. Configure IS-IS protocol to enable dynamic routing protocol to exchange routing information.
3. Configure BGP protocol.
4. Configure segment routing.

To configure flexible algorithm for IS-IS:

1. Define flexible algorithm on routers that you have identified in your network. Assign an ID for the flexible algorithm definition (FAD) ranging from 128 through 255.

```
[edit routing-options]
user@host# set flex-algorithm id
```

> **NOTE**: We recommend configuring flexible algorithm definition on only a few routers to avoid conflicts.

Specify the parameters of the definition. IS-IS calculates the path based on these specified parameters of the FAD.

a. Map a BGP color community to the defined FAD. By default each flexible algorithm is associated with a value equal to the flex algorithm.

VPN can be made to resolve paths over the configured BGP color community.

```
[edit routing-options flex-algorithm id]
user@host# set color desired color community value
```

> **NOTE**: Changing the BGP color community for a flexible algorithm might result in traffic disruption. If you modify a BGP color community for a flexible algorithm then all routes pertaining to that flexible algorithm are removed from the RIB and added again with new colors.

b. Specify the calculation type based on which the IS-IS protocol calculates the path.

```
[edit routing-options flex-algorithm id definition]
user@host# set (spf | strict-spf)
```

c. Specify the metric type based on which IS-IS calculates the path.

```
[edit routing-options flex-algorithm id definition]
user@host# set metric-type (delay-metric | igp-metric | te-metric)
```

d. Assign a priority level to the advertisement of the FAD based on your requirement. Specify a priority ranging from 0 through 255.

```
[edit routing-options flex-algorithm id definition]
user@host# set priority priority
```

NOTE: Modifying the flexible algorithm definition could cause traffic disruptions until all the nodes converge on the new paths.

e. If you have enabled RSVP traffic engineering, you can configure admin-groups for many protocols to color an individual link.

```
[edit protocols mpls]
user@host# set admin-groups
```

f. Define the admin groups as per your requirement.

```
[edit routing-options flex-algorithm definition admin-group]
user@host# set include any admin-group
user@host# set include-all admin-group
user@host# set exclude admin-group
```

NOTE: For FADs with link-constraints to work, all relevant links should advertise the admin-colors in IS-IS. You must either enable RSVP on the interfaces or if you have

not configured RSVP for traffic engineering, make sure you configure **set traffic-engineering advertise always** at the `[edit protocols isis]` hierarchy level.

2. Identify the participating routers and configure participation on those routers. The same device can advertise a FAD and also participate in a flexible algorithm.

```
[edit protocols isis source-packet-routing]
user@host# set flex-algorithm id
```

3. Advertise prefix segments through policy configuration.

```
[edit policy-options policy-statement name term name]
user@host# set from route-filter route exact
user@host# set then prefix-segment algorithm id index value
user@host# set then prefix-segment algorithm id node-segment
```

4. Apply the policy under the protocol IS-IS.

```
[edit protocols isis]
user@host# set export name
```

5. To verify if your flexible algorithm configuration is working correctly use the `show isis spring flex-algorithm` command.

Example: OSPF Flexible Algorithm

IN THIS SECTION

Overview

IN THIS SECTION

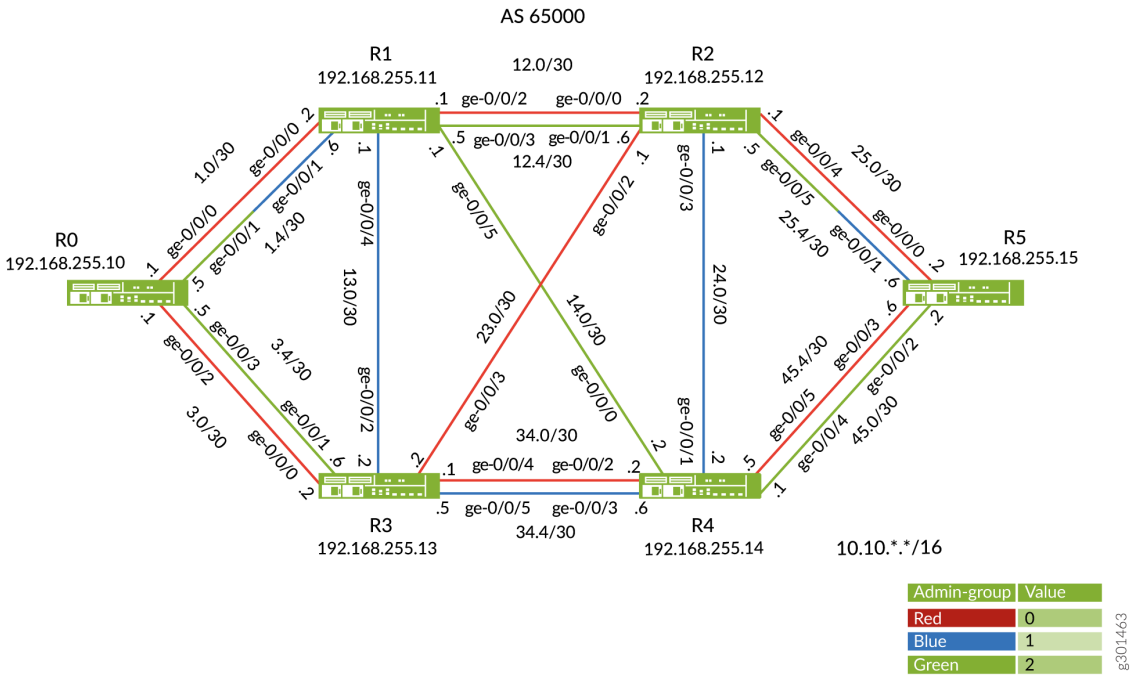
This example shows how to configure flexible algorithm in an OSPFv2 network. The flexible algorithm allows networks without a controller to configure traffic engineering using segment routing without actually implementing a network controller.

You can thin-slice a network by defining flexible algorithms that compute paths using different parameters and link constraints based on your requirements. The set consisting of calculation-type, metric-type, and a set of constraints is referred to as a flexible algorithm definition (FAD). You can define FADs and advertise the same in an OSPFv2 network. A device can also be configured to participate in a certain flexible algorithm provided it supports the constraints for that specific FAD.

Topology

Figure 22 on page 187 shows a flexible algorithm topology in which there are 6 devices R0, R1, R2, R3, R4, and R5. Two flexible algorithms 128 and 129 are defined on each of these devices. The admin-groups red, blue, and green are configured on the devices. The FADs with different parameters such as metric-types, calculation-types, and link constraints are defined on each of the devices.

Figure 22: Flexible Algorithm Topology



Requirements

This example uses the following hardware and software components:

- Six MX Series routers.
- Junos OS Release 21.1R1 or later running on all devices.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Device R0

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description R0_to_R1_1
set interfaces ge-0/0/0 unit 0 family inet address 10.10.1.1/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description R0_to_R1_2
set interfaces ge-0/0/1 unit 0 family inet address 10.10.1.5/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description R0_to_R3_1
set interfaces ge-0/0/2 unit 0 family inet address 10.10.3.1/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description R0_to_R3_2
set interfaces ge-0/0/3 unit 0 family inet address 10.10.3.5/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.10/32
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement ex-bgp term 1 from route-filter 10.1.1.0/24 exact
set policy-options policy-statement ex-bgp term 1 then community add blue
set policy-options policy-statement ex-bgp term 1 then accept
set policy-options policy-statement ex-bgp term 0 from route-filter 10.1.0.0/24 exact
set policy-options policy-statement ex-bgp term 0 then community add red
set policy-options policy-statement ex-bgp term 0 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1001 from route-filter 192.168.255.10/32
exact
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 index
1280
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 index
1290
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 130 index
1300
```

```
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 130 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 131 index
1310
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 131 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 132 index
1320
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 132 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 133 index
1330
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 133 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 134 index
1340
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 134 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 135 index
1350
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 135 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment index 1000
set policy-options policy-statement prefix-sid term 1001 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1001 then accept
set policy-options community blue members color:1:129
set policy-options community red members color:0:128
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls admin-groups GREEN 2
set protocols mpls label-range static-label-range 1000 8000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group RED
set protocols mpls interface ge-0/0/1.0 admin-group GREEN
set protocols mpls interface ge-0/0/2.0 admin-group RED
set protocols mpls interface ge-0/0/3.0 admin-group GREEN
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf traffic-engineering advertisement always
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing srgb start-label 80000
set protocols ospf source-packet-routing srgb index-range 5000
```

```
set protocols ospf source-packet-routing flex-algorithm 128
set protocols ospf source-packet-routing flex-algorithm 129
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 post-convergence-lfa node-protection
set routing-options flex-algorithm 128 definition metric-type igp-metric
set routing-options flex-algorithm 128 definition spf
set routing-options flex-algorithm 128 definition admin-group include-any RED
set routing-options flex-algorithm 129 definition metric-type te-metric
set routing-options flex-algorithm 129 definition spf
set routing-options flex-algorithm 129 definition admin-group include-all BLUE
set routing-options router-id 192.168.255.10
set routing-options autonomous-system 65000
set routing-options static route 10.1.1.0/24 receive
set routing-options static route 10.1.0.0/24 receive
set routing-options forwarding-table export pplb
set routing-options flex-algorithm 128 use-transport-class
set routing-options transport-class auto-create
```

Device R1

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description R1_to_R0_1
set interfaces ge-0/0/0 unit 0 family inet address 10.10.1.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description R1_to_R0_2
set interfaces ge-0/0/1 unit 0 family inet address 10.10.1.6/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description R1_to_R2_1
set interfaces ge-0/0/2 unit 0 family inet address 10.10.12.1/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description R1_to_R2_2
set interfaces ge-0/0/3 unit 0 family inet address 10.10.12.5/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 description R1_to_R3
set interfaces ge-0/0/4 unit 0 family inet address 10.10.13.1/30
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 description R1_to_R4
set interfaces ge-0/0/5 unit 0 family inet address 10.10.14.1/30
```

```
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.11/32
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1001 from route-filter 192.168.255.11/32
exact
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 index
1281
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 index
1291
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment index 1001
set policy-options policy-statement prefix-sid term 1001 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1001 then accept
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls admin-groups GREEN 2
set protocols mpls label-range static-label-range 1000 8000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group RED
set protocols mpls interface ge-0/0/1.0 admin-group GREEN
set protocols mpls interface ge-0/0/2.0 admin-group RED
set protocols mpls interface ge-0/0/3.0 admin-group GREEN
set protocols mpls interface ge-0/0/4.0 admin-group BLUE
set protocols mpls interface ge-0/0/5.0 admin-group GREEN
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf traffic-engineering advertisement always
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing srgb start-label 80000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing flex-algorithm 128
set protocols ospf source-packet-routing flex-algorithm 129
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 post-convergence-lfa node-protection
```

```
set routing-options router-id 192.168.255.11
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
```

Device R2

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description R2_to_R1_1
set interfaces ge-0/0/0 unit 0 family inet address 10.10.12.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description R2_to_R1_2
set interfaces ge-0/0/1 unit 0 family inet address 10.10.12.6/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description R2_to_R3

set interfaces ge-0/0/2 unit 0 family inet address 10.10.23.1/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description R2_to_R4
set interfaces ge-0/0/3 unit 0 family inet address 10.10.24.1/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 description R2_to_R5_1
set interfaces ge-0/0/4 unit 0 family inet address 10.10.25.1/30

set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 description R2_to_R5_2
set interfaces ge-0/0/5 unit 0 family inet address 10.10.25.5/30
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.12/32
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1001 from route-filter 192.168.255.12/32
exact
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 index
1282
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 index
1292
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment index 1002
set policy-options policy-statement prefix-sid term 1001 then prefix-segment node-segment
```

```
set policy-options policy-statement prefix-sid term 1001 then accept
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls admin-groups GREEN 2
set protocols mpls label-range static-label-range 1000 8000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group RED
set protocols mpls interface ge-0/0/1.0 admin-group GREEN
set protocols mpls interface ge-0/0/2.0 admin-group RED
set protocols mpls interface ge-0/0/3.0 admin-group BLUE
set protocols mpls interface ge-0/0/4.0 admin-group RED
set protocols mpls interface ge-0/0/5.0 admin-group GREEN
set protocols mpls interface ge-0/0/5.0 admin-group BLUE
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf traffic-engineering advertisement always
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing srgb start-label 80000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing flex-algorithm 128
set protocols ospf source-packet-routing flex-algorithm 129
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 post-convergence-lfa node-protection
set routing-options router-id 192.168.255.12
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
```

Device R3

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description R3_to_R0_1
set interfaces ge-0/0/0 unit 0 family inet address 10.10.3.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description R3_to_R0_2
set interfaces ge-0/0/1 unit 0 family inet address 10.10.3.6/30
```

```
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description R3_to_R1
set interfaces ge-0/0/2 unit 0 family inet address 10.10.13.2/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description R3_to_R2
set interfaces ge-0/0/3 unit 0 family inet address 10.10.23.2/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 description R3_to_R4_1
set interfaces ge-0/0/4 unit 0 family inet address 10.10.34.1/30
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 description R3_to_R4_2
set interfaces ge-0/0/5 unit 0 family inet address 10.10.34.5/30
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.13/32
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1001 from route-filter 192.168.255.13/32
exact
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 index
1284
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 index
1294
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment index 1003
set policy-options policy-statement prefix-sid term 1001 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1001 then accept
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls admin-groups GREEN 2
set protocols mpls label-range static-label-range 1000 8000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group RED
set protocols mpls interface ge-0/0/1.0 admin-group GREEN
set protocols mpls interface ge-0/0/2.0 admin-group BLUE
set protocols mpls interface ge-0/0/3.0 admin-group RED
set protocols mpls interface ge-0/0/4.0 admin-group RED
set protocols mpls interface ge-0/0/5.0 admin-group BLUE
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf traffic-engineering advertisement always
```

```
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing srgb start-label 80000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing flex-algorithm 128
set protocols ospf source-packet-routing flex-algorithm 129
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 post-convergence-lfa node-protection
set routing-options router-id 192.168.255.13
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
```

Device R4

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description R4_to_R1
set interfaces ge-0/0/0 unit 0 family inet address 10.10.14.2/30

set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description R4_to_R2
set interfaces ge-0/0/1 unit 0 family inet address 10.10.24.2/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description R4_to_R3_1
set interfaces ge-0/0/2 unit 0 family inet address 10.10.34.2/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description R4_to_R3_2
set interfaces ge-0/0/3 unit 0 family inet address 10.10.34.6/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 description R4_to_R5_1
set interfaces ge-0/0/4 unit 0 family inet address 10.10.45.1/30
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 description R4_to_R5_2
set interfaces ge-0/0/5 unit 0 family inet address 10.10.45.5/30
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.14/32
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1001 from route-filter 192.168.255.14/32
```

```
exact
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 index
1284
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 index
1294
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment index 1004
set policy-options policy-statement prefix-sid term 1001 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1001 then accept
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls admin-groups GREEN 2
set protocols mpls label-range static-label-range 1000 8000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/2.0 admin-group RED
set protocols mpls interface ge-0/0/3.0 admin-group BLUE
set protocols mpls interface ge-0/0/0.0 admin-group GREEN
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
set protocols mpls interface ge-0/0/4.0 admin-group GREEN
set protocols mpls interface ge-0/0/5.0 admin-group RED
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf traffic-engineering advertisement always
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing srgb start-label 80000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing flex-algorithm 128
set protocols ospf source-packet-routing flex-algorithm 129
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 post-convergence-lfa node-protection
set routing-options router-id 192.168.255.14
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
```

Device R5

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description R5_to_R2_1
set interfaces ge-0/0/0 unit 0 family inet address 10.10.25.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description R5_to_R2_2
set interfaces ge-0/0/1 unit 0 family inet address 10.10.25.6/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description R5_to_R4_1
set interfaces ge-0/0/2 unit 0 family inet address 10.10.45.2/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description R5_to_R4_2
set interfaces ge-0/0/3 unit 0 family inet address 10.10.45.6/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.255.15/32
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1001 from route-filter 192.168.255.15/32
exact
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 index
1285
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 index
1295
set policy-options policy-statement prefix-sid term 1001 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1001 then prefix-segment index 1005
set policy-options policy-statement prefix-sid term 1001 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1001 then accept
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls admin-groups GREEN 2
set protocols mpls label-range static-label-range 1000 8000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group RED
set protocols mpls interface ge-0/0/1.0 admin-group GREEN
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
set protocols mpls interface ge-0/0/2.0 admin-group GREEN
set protocols mpls interface ge-0/0/3.0 admin-group RED
```

```
set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
set protocols ospf backup-spf-options use-source-packet-routing
set protocols ospf traffic-engineering advertisement always
set protocols ospf source-packet-routing prefix-segment prefix-sid
set protocols ospf source-packet-routing srgb start-label 80000
set protocols ospf source-packet-routing srgb index-range 5000
set protocols ospf source-packet-routing flex-algorithm 128
set protocols ospf source-packet-routing flex-algorithm 129
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 post-convergence-lfa node-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 post-convergence-lfa node-protection
set routing-options router-id 192.168.255.15
set routing-options autonomous-system 65000
set routing-options static route 10.1.15.0/24 reject
set routing-options forwarding-table export pplb
```

Configuring Device R0

To configure flexible algorithm for OSPFv2, perform the following steps on the device R0:

1. Configure the device interfaces to enable IP transport.

```
[edit]
user@R0set interfaces ge-0/0/0 description R0_to_R1_1
user@R0set interfaces ge-0/0/0 unit 0 family inet address 10.10.1.1/30
user@R0set interfaces ge-0/0/0 unit 0 family mpls
user@R0set interfaces ge-0/0/1 description R0_to_R1_2
user@R0set interfaces ge-0/0/1 unit 0 family inet address 10.10.1.5/30
user@R0set interfaces ge-0/0/1 unit 0 family mpls
user@R0set interfaces ge-0/0/2 description R0_to_R3_1
user@R0set interfaces ge-0/0/2 unit 0 family inet address 10.10.3.1/30
user@R0set interfaces ge-0/0/2 unit 0 family mpls
user@R0set interfaces ge-0/0/3 description R0_to_R3_2
user@R0set interfaces ge-0/0/3 unit 0 family inet address 10.10.3.5/30
user@R0set interfaces ge-0/0/3 unit 0 family mpls
```

2. Configure the loopback interface (lo0) address that is used as router ID for OSPF sessions.

```
[edit]
user@R0set interfaces lo0 unit 0 family inet address 192.168.255.10/32
```

3. Configure the router ID and autonomous system (AS) number to propagate routing information within a set of routing devices that belong to the same AS.

```
[edit]
user@R0set routing-options router-id 192.168.255.10
user@R0set routing-options autonomous-system 65000
```

4. Define a policy to load balance packets and apply the per-packet policy to enable load balancing of traffic.

```
[edit]
user@R0set policy-options policy-statement pplb then load-balance per-packet
user@R0set routing-options forwarding-table export pplb
```

5. Configure the route filter for the routing policy term that enables the Device R0 to reach the 192.168.255.10/32 network.

```
[edit]
user@R0set policy-options policy-statement prefix-sid term 1001 from route-filter
192.168.255.10/32 exact
```

6. Configure MPLS on all interfaces excluding the management interface.

```
[edit]
user@R0set protocols mpls interface all
user@R0set protocols mpls interface fxp0.0 disable
```

7. Configure the MPLS label range to assign static labels for the links.

```
[edit]
user@R0set protocols mpls label-range static-label-range 1000 8000
```

8. Configure TI-LFA to enable protection against link and node failures. SR using TI-LFA provides faster restoration of network connectivity by routing the traffic instantly to a backup or an alternate path if the primary path fails or becomes unavailable.

```
[edit]
user@R0set protocols ospf backup-spf-options use-source-packet-routing
```

9. Configure the maximum number of labels for segment routing routed paths for protection of backup shortest-path-first attributes.

```
[edit]
user@R0set protocols ospf backup-spf-options use-post-convergence-lfa maximum-labels 5
```

10. Configure prefix segment attributes, the start label and the index range for segment routing global blocks (SRGBs) in SPRING for the OSPF protocol.

```
[edit]
user@R0set protocols ospf source-packet-routing prefix-segment prefix-sid
user@R0set protocols ospf source-packet-routing srgb start-label 80000
user@R0set protocols ospf source-packet-routing srgb index-range 5000
```

11. Enable node-link protection on the OSPF interfaces that follow post-convergence path.

```
[edit]
user@R0set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 post-convergence-lfa node-
protection
user@R0set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 post-convergence-lfa node-
protection
```

```
user@R0set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 post-convergence-lfa node-
protection
user@R0set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 post-convergence-lfa node-
protection
```

12. Configure the loopback interface as passive to ensure the protocols do not run over the loopback interface and that the loopback interface is advertised correctly throughout the network.

```
[edit]
user@R0set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

13. Define flexible algorithms on the device R0. Assign a name for each of the FADs ranging from 128 through 255.

```
[edit]
user@R0set routing-options flex-algorithm 128
user@R0set routing-options flex-algorithm 129
```

Specify the parameters of the definition. OSPFv2 calculates the path based on these specified parameters of the FAD.

a. Specify the calculation type based on which the OSPFv2 protocol calculates the path.

```
[edit]
user@R0set routing-options flex-algorithm 128 definition spf
user@R0set routing-options flex-algorithm 128 definition spf
```

b. Specify the metric type based on which OSPFv2 calculates the path.

```
[edit]
user@R0set routing-options flex-algorithm 128 definition metric-type igp-metric
user@R0set routing-options flex-algorithm 129 definition metric-type te-metric
```

c. If you have enabled RSVP traffic engineering, you can configure admin-groups for many protocols to color an individual link.

```
[edit]
user@R0set protocols mpls admin-groups RED 0
user@R0set protocols mpls admin-groups BLUE 1
user@R0set protocols mpls admin-groups GREEN 2
```

d. Assign the configured admin-groups policies to the device R0 interfaces.

```
[edit]
user@R0set protocols mpls interface ge-0/0/0.0 admin-group RED
user@R0set protocols mpls interface ge-0/0/1.0 admin-group GREEN
user@R0set protocols mpls interface ge-0/0/2.0 admin-group RED
user@R0set protocols mpls interface ge-0/0/3.0 admin-group GREEN
```

e. Define the admin-groups as per your requirement.

```
[edit]
user@R0set routing-options flex-algorithm 128 definition admin-group include-any RED
user@R0set routing-options flex-algorithm 129 definition admin-group include-all GREEN
user@R0set routing-options flex-algorithm 129 definition admin-group include-all BLUE
```

NOTE: For FADs with link-constraints to work, all relevant links should advertise the admin-colors in OSPFv2. You must either enable RSVP on the interfaces or if you have not configured RSVP for traffic engineering, make sure you configure set traffic-engineering advertise always at the `[edit protocols ospf]` hierarchy level.

```
[edit]
user@R0set protocols ospf traffic-engineering advertisement always
```

14. Configure the flexible algorithm participation on the device R0. The same device can advertise a FAD and also participate in a flexible algorithm.

```
[edit]
user@R0set protocols ospf source-packet-routing flex-algorithm 128
user@R0set protocols ospf source-packet-routing flex-algorithm 129
```

15. Advertise prefix segments through policy configuration.

```
[edit]
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 128 index 1280
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 128 node-segment
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 129 index 1290
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 129 node-segment
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 130 index 1300
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 130 node-segment
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 131 index 1310
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 131 node-segment
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 132 index 1320
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 132 node-segment
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 133 index 1330
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 133 node-segment
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 134 index 1340
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 134 node-segment
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
```

```
algorithm 135 index 1350
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment
algorithm 135 node-segment
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment index
1000
user@R0set policy-options policy-statement prefix-sid term 1001 then prefix-segment node-
segment
user@R0set policy-options policy-statement prefix-sid term 1001 then accept
```

Results

Check the results of the configuration:

From configuration mode, confirm your configuration by entering the, `show interfaces`, `show routing-options`, `show protocols`, and `show policy-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
interfaces {
    ge-0/0/0 {
        description R0_to_R1_1;
        unit 0 {
            family inet {
                address 10.10.1.1/30;
            }
            family mpls;
        }
    }
    ge-0/0/1 {
        description R0_to_R1_2
        unit 0 {
            family inet {
                address 10.10.1.5/30;
            }
            family mpls;
          }
    }
    ge-0/0/2 {
        description R0_to_R3_1
        unit 0 {
            family inet {
                address 10.10.3.1/30;
            }
```

```
            family mpls;
        }
    }
    ge-0/0/3 {
        description R0_to_R3_2
        unit 0 {
            family inet {
                        address 10.10.3.5/30;
                    }
                    family mpls;
             }
     }

    lo0 {
        unit 0 {
            family inet {
                address 192.168.255.10/32;
            }
        }
    }
}
policy-options {
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
    policy-statement prefix-sid {
        term 1001 {
            from {
                route-filter 192.168.255.10/32 exact;
            }
            then {
                prefix-segment {
                    algorithm 128 index 1280 node-segment;
                    algorithm 129 index 1290 node-segment;
                    algorithm 130 index 1300 node-segment;
                    algorithm 131 index 1310 node-segment;
                    algorithm 132 index 1320 node-segment;
                    algorithm 133 index 1330 node-segment;
                    algorithm 134 index 1340 node-segment;
                    algorithm 135 index 1350 node-segment;
                    index 1000;
```

```
                    node-segment;
                }
                accept;
            }
        }
    }
}
protocols {
    mpls {
        admin-groups {
            RED 0;
            BLUE 1;
            GREEN 2;
        }
        label-range {
            static-label-range 1000 8000;
        }
        interface all;
        interface fxp0.0 {
            disable;
        }
        interface ge-0/0/0.0 {
            admin-group RED;
        }
        interface ge-0/0/1.0 {
            admin-group GREEN;
        }
        interface ge-0/0/2.0 {
            admin-group RED;
        }
        interface ge-0/0/3.0 {
            admin-group GREEN;
        }
    }
    ospf {
        backup-spf-options {
            use-post-convergence-lfa maximum-labels 5;
            use-source-packet-routing;
        }
        traffic-engineering {
            advertisement always;
        }
        source-packet-routing {
```

```
            prefix-segment prefix-sid;
            srgb start-label 80000 index-range 5000;
            flex-algorithm [ 128 129 ];
        }
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface ge-0/0/0.0 {
                post-convergence-lfa {
                    node-protection;
                }
            }
            interface ge-0/0/1.0 {
                post-convergence-lfa {
                    node-protection;
                }
            }
            interface ge-0/0/2.0 {
                post-convergence-lfa {
                    node-protection;
                }
            }
            interface ge-0/0/3.0 {
                post-convergence-lfa {
                    node-protection;
                }
            }
        }
    }
}
routing-options {
    flex-algorithm 128 {
        definition {
            metric-type igp-metric;
            spf;
            admin-group include-any RED;
        }
    }
    flex-algorithm 129 {
        definition {
            metric-type te-metric;
            spf;
```

```
            admin-group include-all [ GREEN BLUE ];
        }
    }
    router-id 192.168.255.10;
    autonomous-system 65000;
    forwarding-table {
        export pplb;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform the following tasks:

Verifying the OSPF Database

Purpose

Verifying that the flexible algorithm signaling is displayed in the OSPF database.

Action

From operational mode, run the `show ospf database opaque-area extensive` command.

On R0

```
user@R0>show ospf database opaque-area extensive

OpaqArea 4.0.0.0          192.168.255.10   0x800000ad   503  0x22 0xb85d  76
Opaque LSA
  SR-Algorithm (8), length 3:
    Algo (1), length 1:
        0
    Algo (2), length 1:
        128
    Algo (3), length 1:
        129
  SID/Label Range (9), length 12:
    Range Size (1), length 3:
        5000
    SID/Label (1), length 3:
      Label (1), length 3:
        80000
  Flex-Algorithm Definition (16), length 12:
      Flex-Algo (1), length 1:
        128
      Metric-Type (2), length 1:
        0
      Calc-Type (3), length 1:
        0
      Priority (4),  length 1:
        0
     FAD AG Include Any (2), length 4:
        Include Any AG (1), length 4:
        0x1
  Flex-Algorithm Definition (16), length 12:
      Flex-Algo (1), length 1:
        129
      Metric-Type (2), length 1:
        2
      Calc-Type (3), length 1:
        0
      Priority (4),  length 1:
        0
     FAD AG Include All (3), length 4:
        Include All AG (1), length 4:
```

```
        0x6
  Aging timer 00:51:37
  Installed 00:08:20 ago, expires in 00:51:37, sent 00:08:18 ago
  Last changed 5d 13:35:52 ago, Change count:
```

Meaning

This output on R0 illustrates that:

Three segment-routing algorithms (including two flexible algorithms) are advertised by this device.

Two FADs are advertised by this device.

Verifying the Flexible Algorithm Details

Purpose

Verifying that the flexible algorithm details are displayed.

Action

From operational mode, run the `show ospf spring flex-algorithm <flex-algorithm-id>` command.

On R0

```
user@R0>show ospf spring flex-algorithm 128
Flex Algo: 128, Area: 0.0.0.0
  Color: 128, Participating, FAD supported
      Winner: 192.168.255.10, Metric: 0, Calc: 0, Prio: 0, inc-any: 0x1, FAD supported
        Include-Any:  0x1 RED
  SPF Version: 296
  Participation toggles: 1
  Topo refresh count: 0
  Full SPFs: 296, Partial SPFs: 0
```

Meaning

The flexible algorithm details that are configured on R0 are displayed.

Verifying Flexible Algorithm Specific OSPF Internal Routes

Purpose

Verifying that the flexible algorithm specific OSPF internal routes are displayed.

Action

From operational mode, run the `show ospf route flex-algorithm <flex-algorithm-id>` command.

On R0

```
user@R0>show ospf route flex-algorithm 128
Prefix             Path  Route      NH       Metric NextHop       Nexthop
                   Type  Type       Type             Interface     Address/LSP
192.168.255.11     Intra Router     IP            1 ge-0/0/0.0    10.10.1.2
                                                     ge-0/0/1.0    10.10.1.6
192.168.255.12     Intra Router     IP            2 ge-0/0/0.0    10.10.1.2
                                                     ge-0/0/1.0    10.10.1.6
                                                     ge-0/0/2.0    10.10.3.2
                                                     ge-0/0/3.0    10.10.3.6
192.168.255.13     Intra Router     IP            1 ge-0/0/2.0    10.10.3.2
                                                     ge-0/0/3.0    10.10.3.6
192.168.255.14     Intra Router     IP            2 ge-0/0/0.0    10.10.1.2
                                                     ge-0/0/1.0    10.10.1.6
                                                     ge-0/0/2.0    10.10.3.2
                                                     ge-0/0/3.0    10.10.3.6
192.168.255.15     Intra Router     IP            3 ge-0/0/0.0    10.10.1.2
                                                     ge-0/0/1.0    10.10.1.6
                                                     ge-0/0/2.0    10.10.3.2
                                                     ge-0/0/3.0    10.10.3.6
10.10.1.0/30       Intra Network    IP            1 ge-0/0/0.0
10.10.1.4/30       Intra Network    IP            1 ge-0/0/1.0
10.10.3.0/30       Intra Network    IP            1 ge-0/0/2.0
10.10.3.4/30       Intra Network    IP            1 ge-0/0/3.0
10.10.12.0/30      Intra Network    IP            2 ge-0/0/0.0    10.10.1.2
                                                     ge-0/0/1.0    10.10.1.6
10.10.12.4/30      Intra Network    IP            2 ge-0/0/0.0    10.10.1.2
                                                     ge-0/0/1.0    10.10.1.6
10.10.13.0/30      Intra Network    IP            2 ge-0/0/0.0    10.10.1.2
                                                     ge-0/0/1.0    10.10.1.6
                                                     ge-0/0/2.0    10.10.3.2
```

```
                                                ge-0/0/3.0    10.10.3.6
10.10.14.0/30      Intra Network    IP          2 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
10.10.23.0/30      Intra Network    IP          2 ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
10.10.24.0/30      Intra Network    IP          3 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
10.10.25.0/30      Intra Network    IP          3 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
10.10.25.4/30      Intra Network    IP          3 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
10.10.34.0/30      Intra Network    IP          2 ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
10.10.34.4/30      Intra Network    IP          2 ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
10.10.45.0/30      Intra Network    IP          3 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
10.10.45.4/30      Intra Network    IP          3 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
128.49.106.245/32  Intra Network    IP          1 ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
128.49.107.40/32   Intra Network    IP          2 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
192.168.255.10/32  Intra Network    IP          0 lo0.0
192.168.255.10/32  Intra Network    Spring      0 lo0.0
192.168.255.11/32  Intra Network    IP          1 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
192.168.255.11/32  Intra Network    Spring      1 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
192.168.255.12/32  Intra Network    IP          2 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
```

```
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
192.168.255.12/32  Intra Network    Spring        2 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
192.168.255.13/32  Intra Network    IP            1 ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
192.168.255.13/32  Intra Network    Spring        1 ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
192.168.255.14/32  Intra Network    IP            2 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
192.168.255.14/32  Intra Network    Spring        2 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
192.168.255.15/32  Intra Network    IP            3 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
192.168.255.15/32  Intra Network    Spring        3 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
16                 Intra Network    Mpls          0 ge-0/0/0.0    10.10.1.2
                                    Bkup MPLS       ge-0/0/1.0    10.10.1.6
16 (S=0)           Intra Network    Mpls          0 ge-0/0/0.0    10.10.1.2
                                    Bkup MPLS       ge-0/0/1.0    10.10.1.6
17                 Intra Network    Mpls          0 ge-0/0/1.0    10.10.1.6
                                    Bkup MPLS       ge-0/0/0.0    10.10.1.2
17 (S=0)           Intra Network    Mpls          0 ge-0/0/1.0    10.10.1.6
                                    Bkup MPLS       ge-0/0/0.0    10.10.1.2
20                 Intra Network    Mpls          0 ge-0/0/2.0    10.10.3.2
                                    Bkup MPLS       ge-0/0/3.0    10.10.3.6
20 (S=0)           Intra Network    Mpls          0 ge-0/0/2.0    10.10.3.2
                                    Bkup MPLS       ge-0/0/3.0    10.10.3.6
21                 Intra Network    Mpls          0 ge-0/0/3.0    10.10.3.6
                                    Bkup MPLS       ge-0/0/2.0    10.10.3.2
21 (S=0)           Intra Network    Mpls          0 ge-0/0/3.0    10.10.3.6
                                    Bkup MPLS       ge-0/0/2.0    10.10.3.2
81001              Intra Network    Mpls          1 ge-0/0/0.0    10.10.1.2
```

```
                                                ge-0/0/1.0    10.10.1.6
81001 (S=0)        Intra Network    Mpls        1 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
81003              Intra Network    Mpls        2 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
81004              Intra Network    Mpls        1 ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
81004 (S=0)        Intra Network    Mpls        1 ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
81006              Intra Network    Mpls        2 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
81007              Intra Network    Mpls        3 ge-0/0/0.0    10.10.1.2
                                                ge-0/0/1.0    10.10.1.6
                                                ge-0/0/2.0    10.10.3.2
                                                ge-0/0/3.0    10.10.3.6
```

Meaning

The `show ospf route` command is extended with `flex-algorithm` option to show flexible algorithm specific OSPF internal routes. Each route is prefixed with the *flex-algo-id*.

Verifying Flex Colored routes

Purpose

Verifying that the flexible algorithm specific OSPF internal routes are displayed.

Action

From operational mode, run the `show route protocol ospf` command.

On R0

```
user@R0>show route protocol ospf
junos-rti-tc-<color>.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.255.11-128<c>/64
                   *[L-OSPF/10/5] 1w2d 01:23:04, metric 1
                    >  to 10.10.1.2 via ge-0/0/0.0
                       to 10.10.3.2 via ge-0/0/2.0, Push 81281, Push 81283(top)
192.168.255.12-128<c>/64
                   *[L-OSPF/10/5] 1w2d 01:23:04, metric 2
                       to 10.10.1.2 via ge-0/0/0.0, Push 81283
                    >  to 10.10.3.2 via ge-0/0/2.0, Push 81283
192.168.255.13-128<c>/64
                   *[L-OSPF/10/5] 1w2d 01:23:04, metric 1
                    >  to 10.10.3.2 via ge-0/0/2.0
                       to 10.10.1.2 via ge-0/0/0.0, Push 81284, Push 81283(top)
192.168.255.14-128<c>/64
                   *[L-OSPF/10/5] 1w2d 01:23:04, metric 2
                    >  to 10.10.3.2 via ge-0/0/2.0, Push 81286
                       to 10.10.1.2 via ge-0/0/0.0, Push 81286, Push 81283(top)
192.168.255.15-128<c>/64
                   *[L-OSPF/10/5] 1w2d 01:23:04, metric 3
                       to 10.10.1.2 via ge-0/0/0.0, Push 81287
                    >  to 10.10.3.2 via ge-0/0/2.0, Push 81287
```

Meaning

The output displays all the colored flex routes programmed in junos-rti-tc-<color>.inet.0 table in the following format: *prefix_address-flex-algo-id*<c>/64

Verifying OSPF Logs

Purpose

Verifying that the OSPF logs displays the flexible algorithm keyword.

Action

From operational mode, run the `show ospf log` command.

On R0

```
user@R0>show ospf log
Topology default SPF log:
```

```
   Last instance of each event type
When            Type            Elapsed
1w2d 13:59:18   SPF             0.000316
1w2d 13:59:18   Stub            0.000233
1w2d 13:59:18   Interarea       0.000002
1w2d 13:59:18   External        0.000004
1w2d 13:59:18   NSSA            0.000001
1w2d 13:59:18   Cleanup         0.000551

   Maximum length of each event type
When            Type            Elapsed
1w2d 14:34:27   SPF             0.000997
1w2d 15:59:35   Stub            0.000675
1w3d 07:08:27   Interarea       0.000010
1w3d 07:29:07   External        0.000013
1w3d 07:15:21   NSSA            0.000008
1w3d 08:38:05   Cleanup         0.001044

   Last 100 events
When            Type            Elapsed

1w2d 14:08:36   FlexAlgo SPF    0.000680
1w2d 14:08:36   SPF             0.000204
1w2d 14:08:36   Stub            0.000025
1w2d 14:08:36   Interarea       0.000003
1w2d 14:08:36   External        0.000002
1w2d 14:08:36   NSSA            0.000001
1w2d 14:08:36   Prefix SID      0.000222
1w2d 14:08:36   Adj SID         0.000074
1w2d 14:08:36   Cleanup         0.000607
1w2d 14:08:36     Total         0.001209
1w2d 14:08:31   SPF             0.000188
1w2d 14:08:31   Stub            0.000054
1w2d 14:08:31   Interarea       0.000002
1w2d 14:08:31   External        0.000001
1w2d 14:08:31   NSSA            0.000001
1w2d 14:08:31   Prefix SID      0.000181
1w2d 14:08:31   Adj SID         0.000178
1w2d 14:08:31   Cleanup         0.000413
1w2d 14:08:31     Total         0.001656
1w2d 14:06:54   FlexAlgo SPF    0.001914
1w2d 14:06:54   FlexAlgo SPF    0.000081
1w2d 14:06:54   SPF             0.000215
```

```
1w2d 14:06:54   Stub            0.000030
1w2d 14:06:54   Interarea       0.000003
1w2d 14:06:54   External        0.000001
1w2d 14:06:54   NSSA            0.000001
1w2d 14:06:54   Prefix SID      0.000227
1w2d 14:06:54   Adj SID         0.000075
1w2d 14:06:54   Cleanup         0.000233
1w2d 14:06:54     Total         0.000859
1w2d 14:06:49   SPF             0.000234
1w2d 14:06:49   Stub            0.000072
1w2d 14:06:49   Interarea       0.000003
1w2d 14:06:49   External        0.000002
1w2d 14:06:49   NSSA            0
1w2d 14:06:49   Prefix SID      0.000262
1w2d 14:06:49   Adj SID         0.000254
1w2d 14:06:49   Cleanup         0.000495
1w2d 14:06:49     Total         0.001936
1w2d 14:06:30   FlexAlgo SPF    0.001356
1w2d 14:06:30   FlexAlgo SPF    0.000061
1w2d 14:06:30   SPF             0.000207
1w2d 14:06:30   Stub            0.000023
1w2d 14:06:30   Interarea       0.000003
1w2d 14:06:30   External        0.000002
1w2d 14:06:30   NSSA            0.000001
1w2d 14:06:30   Prefix SID      0.000237
1w2d 14:06:30   Adj SID         0.000087
1w2d 14:06:30   Cleanup         0.000430
1w2d 14:06:30     Total         0.001060
1w2d 14:06:25   SPF             0.000207
1w2d 14:06:25   Stub            0.000077
1w2d 14:06:25   Interarea       0.000002
1w2d 14:06:25   External        0.000002
1w2d 14:06:25   NSSA            0.000001
1w2d 14:06:25   Prefix SID      0.000250
1w2d 14:06:25   Adj SID         0.000245
1w2d 14:06:25   Cleanup         0.000399
1w2d 14:06:25     Total         0.001840
1w2d 14:05:56   FlexAlgo SPF    0.001781
1w2d 14:05:56   FlexAlgo SPF    0.000080
1w2d 14:05:55   SPF             0.000215
1w2d 14:05:55   Stub            0.000025
1w2d 14:05:55   Interarea       0.000002
1w2d 14:05:55   External        0.000001
```

```
1w2d 14:05:55   NSSA            0.000001
1w2d 14:05:55   Prefix SID      0.000240
1w2d 14:05:55   Adj SID         0.000073
1w2d 14:05:55   Cleanup         0.000422
1w2d 14:05:55     Total         0.001055
1w2d 14:05:50   SPF             0.000212
1w2d 14:05:50   Stub            0.000082
1w2d 14:05:50   Interarea       0.000003
1w2d 14:05:50   External        0.000001
1w2d 14:05:50   NSSA            0.000001
1w2d 14:05:50   Prefix SID      0.000264
1w2d 14:05:50   Adj SID         0.000239
1w2d 14:05:50   Cleanup         0.000458
1w2d 14:05:50     Total         0.002053
1w2d 13:59:23   FlexAlgo SPF    0.001603
1w2d 13:59:23   FlexAlgo SPF    0.000062
1w2d 13:59:23   SPF             0.000224
1w2d 13:59:23   Stub            0.000021
1w2d 13:59:23   Interarea       0.000002
1w2d 13:59:23   External        0.000001
1w2d 13:59:23   NSSA            0.000001
1w2d 13:59:23   Prefix SID      0.000222
1w2d 13:59:23   Adj SID         0.000087
1w2d 13:59:23   Cleanup         0.000413
1w2d 13:59:23     Total         0.001228
1w2d 13:59:18   SPF             0.000316
1w2d 13:59:18   Stub            0.000233
1w2d 13:59:18   Interarea       0.000002
1w2d 13:59:18   External        0.000004
1w2d 13:59:18   NSSA            0.000001
1w2d 13:59:18   Prefix SID      0.000324
1w2d 13:59:18   Adj SID         0.000318
1w2d 13:59:18   Cleanup         0.000551
1w2d 13:59:18     Total         0.002751
```

Meaning

The output displays the FlexAlgo keyword added for the SPF logs.

Configuring Application-Specific Link Attribute on an OSPF Interface

Advertise different te-attributes such as te-metric, delay-metric, or admin-groups for RSVP and flexible algorithms on the same link. This is done using flexible algorithm specific application-specific link attribute as defined in RFC 8920.

To configure application-specific link attribute based flexible algorithm on an OSPF Interface:

1. Create an OSPF area.

```
[edit protocols]
user@host#set protocols ospf area area-id
```

For example:

```
[edit protocols]
user@host#set protocols ospf area 0.0.0.0
```

2. Specify the interface.

```
[edit protocols ospf area 0.0.0.0]
user@host#set interface interface-name
```

For example:

```
[edit protocols ospf area 0.0.0.0]
user@host#set interface ge-0/0/0.0
```

3. Configure application-specific link attribute on the OSPF interface of the device.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set application-specific
```

4. Specify the attribute group.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific]
user@host#set attribute-group name
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific]
user@host#set attribute-group asla
```

5. Configure flexible algorithm specific te-attributes such as te-metric, delay-metric, and admin-groups. Specify the te-metric for the attribute group. The te-metric indicates the metric type based on which OSPFv2 calculates the path.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific attribute-group
asla]
user@host#set te-metric
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific]
user@host#set 15
```

6. Specify the admin-group for the attribute group.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific attribute-group
asla]
user@host#set admin-group
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific]
user@host#set green
```

7. Specify delay-metric for the attribute group.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific attribute-group
asla]
user@host#set delay-metric
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific]
user@host#set 123123
```

8. In case delay-metric is not configured, specify advertise-interface-delay to fetch the delay values from the interface configuration hierarchy, that is legacy delay values.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific attribute-group
asla]
user@host#set advertise-interface-delay
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific]
user@host#set 123125
```

NOTE: The following configuration can be committed only if all of the following criteria match:

- An application is associated with the attribute group.
- Delay-metric is not configured in the hierarchy.
- Interface level delay configurations are present.

9. Specify the application for the attribute group. In the current implementation, only flexible algorithm can be configured as an application. An attribute group can have more than one applications associated with it and it equates to a single application-specific link attribute with the

application bits set in the standard application identifier bit mask field of the application-specific link attribute sub.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific attribute-group
asla]
user@host#set application aplication-name
```

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0 application-specific attribute-group
asla]
user@host#set application flex-algorithm
```

10. Enter `commit` from the configuration mode.
11. Specify strict-asla-based-flex-algorithm to mandate that flexible algorithm path computations use only the links which advertise relevant te-attributes through application-specific link attribute.

```
[edit protocols ospf source-packet-routing]
user@host#set strict-asla-based-flex-algorithm
```

12. Enter `commit` from the configuration mode.

    To verify your configuration results, use the `show protocols` operational command.

```
ospf {
          area 0.0.0.0
             interface ge-0/0/0.0 {
                    application-specific {
                          attribute-group asla {
                                     te-metric 15;
                                     admin-group green;
                                      delay-metric 123123;
                                      advertise-interface-delay;
                                      application flex-algorithm;

                           }
                    }
            }
          source-packet-routing {
          strict-asla-based-flex-algorithm;
      }
```

```
}
```

The Junos OS and Junos OS Evolved implementation supports application-specific link attribute subTLV to comply with RFC 8920. The application-specific link attribute sub-TLV is a sub-TLV of the OSPFv2 extended Link TLV as defined in RFC 7684.

To verify the presence of application-specific link attribute sub-TLVs in the OSPF database use the `show ospf database extensive` operational command.

```
user@host> show ospf database advertising-router self extensive lsa-id 10.0.0.2
```

```
OSPF database, Area 0.0.0.0
 Type       ID               Adv Rtr           Seq      Age  Opt  Cksum  Len
OpaqArea*10.0.0.2          100.100.100.100  0x80000007   665  0x22 0x649d 104
Opaque LSA
  Extended Link (1), length 80:
    Link Type (1), length 1:
     1
     Link Id (2), length 4:
     10.1.1.1
     Link Data (3), length 4:
     10.21.1.1
    Adjacency Sid (2), length 7:
        Flags (1), length 1:
          0x60
        MT ID (2), length 1:
           0
        Weight (3), length 1:
           0
        Label (4), length 3:
          17
    Application Specific Link Attribute (10), length 52:
        SABM Length (1), length 1:
          4
        UDABM Length (2), length 1:
          0
        SABM (3), length 4:
          0x10
        UDABM (4), length 0:
          0x0
```

```
    TEMetric (5), length 4:
      10
    UnidirecLinkDelay (27), length 4:
      123
    MinMaxUnidirecLinkDelay (28), length 8:
      Min DM: 123, Max DM: 123
    UnidirecLinkDelayVar (29), length 4:
      0
    Color (9), length 4:
      2
  Gen timer 00:34:55
  Aging timer 00:48:55
  Installed 00:11:05 ago, expires in 00:48:55, sent 00:11:05 ago
  Last changed 00:11:05 ago, Change count: 6, Ours, TE Link ID: 0
```

The output displays application-specific link attribute sub-TLV fields and attributes.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
22.2R1	Starting in Junos OS and Junos OS Evolved Release 22.2R1, you can advertise different te-attributes such as te-metric, delay-metric, or admin-groups for RSVP and flexible algorithms on the same link. This is done using flexible algorithm specific application-specific link attribute as defined in RFC 8920.

RELATED DOCUMENTATION

application-specific (Protocols OSPF)

How to Enable Link Delay Measurement and Advertising in IGP

IN THIS SECTION

Understanding Link Delay Measurement and Advertising in IGP

IN THIS SECTION

Benefits of link delay measurement and advertising in IS-IS

Link delay measurement and advertising in IS-IS provides the following benefits:

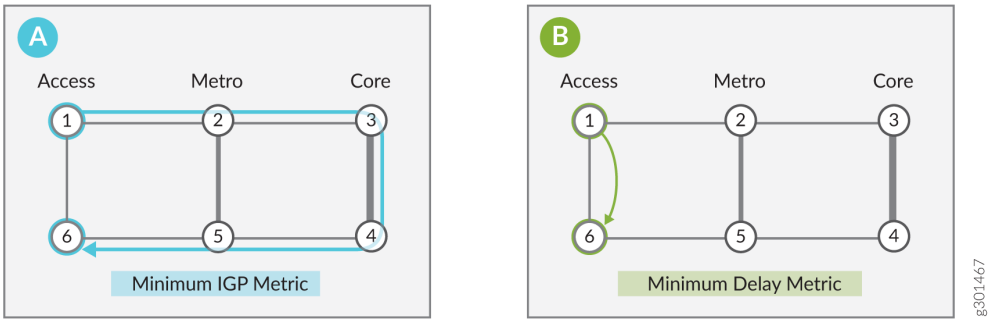
- Highly beneficial in certain networks such as stock market data providers, where it is crucial to have access to market data in real-time to make trades faster than the competition. This is where network performance criteria or latency is becoming critical to data-path selection.
- Helps to make path-selection decisions based on performance data (such as latency) in a cost-effective and scalable way.
- Superior alternative to using metrics such as hop count or cost as routing metrics.

Overview of link delay measurement and advertising in IGP

Network performance is measured by using TWAMP -Light. You can get the measurement of various performance metrics in IP networks, by using probe messages. IS-IS Traffic Engineering Extensions helps to distribute network-performance information in a scalable fashion. This information can then be used to make path-selection decisions based on network performance.

Border Gateway Protocol Link-State (BGP-LS) allows BGP to carry link-state information acquired from IGPs, which then allows internet service providers (ISP) to selectively expose the information with other ISPs, service providers, CDNs and so on, through normal BGP peering. New BGP-Link State (BGP-LS) TLVs are defined to carry the IGP Traffic Engineering Metric Extensions.

The following illustration depicts the minimum IGP metric and minimum delay metric in networks that consist a core, metro, and access network.



In this scenario, core network is cheaper but has longer delay. Access shortcut, with lowest latency is expensive. As core network is cheaper, majority of traffic typically go from 1>2>3>4>5> to 6 by using minimum IGP metric. As displayed in scenario a), you can achieve minimum IGP requirement by running IS-IS with appropriate cost configured and default IS-IS algorithm set to zero. In businesses where ultra-low latency is crucial, packets need to go from 1 to 6. As displayed in scenario b), you can achieve minimum delay metric by defining IS-IS flex algorithm with minimum latency, which minimize the delay to the endpoint. This flex algorithm consists only node 1 and node 6.

Overview of Link Delay Normalization

The Delay Normalization feature computes a normalized delay value and uses the normalized value instead of measured delay value. This value is advertised and used as a metric during the Flexible Algorithm computation. The normalization is performed when the delay is received from the delay measurement component. When the next value is received, it is normalized and compared to the previously saved normalized value. If the values are different, then the LSP generation is triggered.

Delay normalization is disabled by default. To enable and configure delay normalization across IGP instances, use the delay-measurement command.

```
set protocols isis interface interface-name delay-measurement normalize interval value offset
value
```

```
set protocols ospf area aread interface interface-name delay-measurement normalize interval
value offset value
```

SEE ALSO

delay-measurement (Protocols IS-IS)

delay-measurement (Protocols OSPF)

delay-metric (Protocols OSPF)

Understand Two-Way Active Measurement Protocol

Example: Enable IS-IS Link Delay with Source Packet Routing in Networking (SPRING) in a Layer 3 Virtual Private Network (VPN)

IN THIS SECTION

This example shows how to configure IS-IS link delay with SPRING in a Layer3 VPN scenario. In the example, you can create two VPNs between PE1 and PE2. VPN1 optimizes link delay and VPN2 optimizes IGP metric. Although you can configure the feature to enable bidirectional traffic in the test topology, we're focusing on a unidirectional traffic scenario in this example. Specifically, your task is to control the forwarding path for Layer 3 VPN traffic sent by PE1 to destinations advertised by PE2.

Requirements

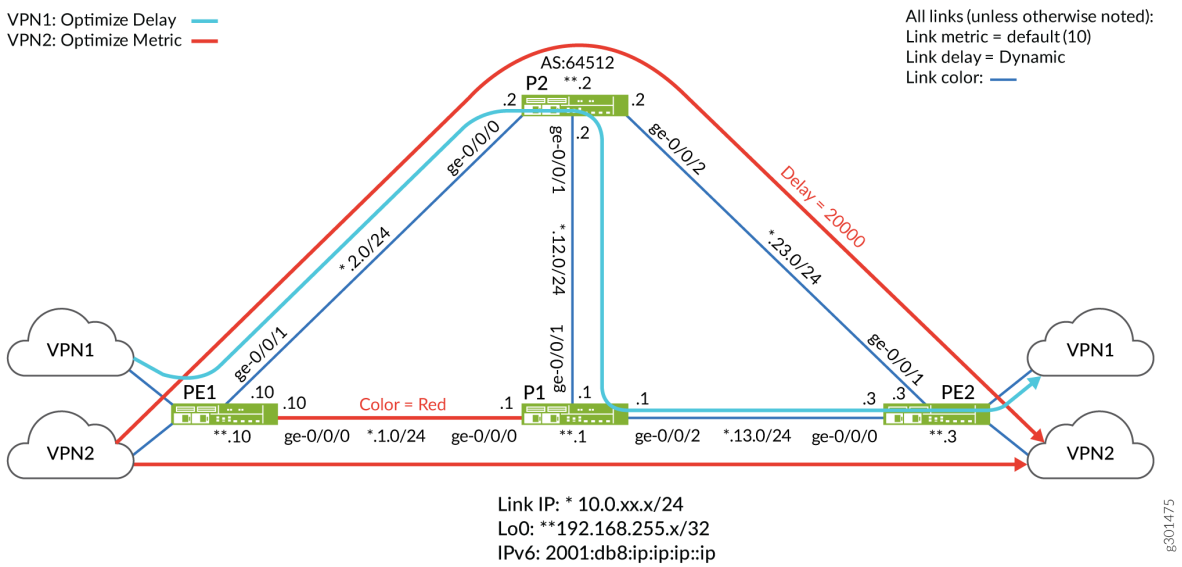
IN THIS SECTION

This example uses the following hardware and software components:

- Four MX Series routers
- Junos OS Release 21.1R1 or later running on all devices

Topology

Figure 23: IS-IS Link Delay Topology



In the topology, most links have a (default) IGP metric of 10, dynamic delay measurements, and blue coloring. The exceptions are the red colored path between PE1 and P1 and the static delay configuration on the P2 to PE2 link.

We've configured the test topology to support IS-IS link delay for both IPv4 and IPv6. We've configured the P2 router as a route reflector with the PE devices as its clients. To keep the topology simple, we are using static routes in the PE2 router's VRFs. This eliminates the need for CE devices and a PE-CE routing protocol such as EBGP.

Your goal is to configure the network so that the routes advertised by PE2 for VPN1 take a path that optimizes delay while also being confined to using only blue links. In contrast, traffic sent to the routes associated with VPN2 can take either a blue or red link with path optimization based on its IGP metric.

- The Flex Algorithm Definition (FAD) for VPN1 uses algorithm 128. We've configured it to use blue colored links only (PE1>P2>P1>PE2) over a path that is optimized to reduce delay. To help demonstrate proper path selection, you configure a static delay of 20000 microseconds between P2 and PE2. This delay is significantly higher than the dynamic delay measured on the remaining links. As a result you expect flex algorithm 128 traffic to avoid the P2 to PE2 link, instead preferring additional hops along the blue color path (PE1>P2>P1>PE2).
- The Flex Algorithm Definition (FAD) for VPN2 uses algorithm 129. We've configured it to take either blue or red links (PE1>P1>PE2 or PE1>P2>PE2), with the path optimized on IGP metric. As a result

traffic using flex algorithm 129 has two equal cost paths between PE1 and PE2, both incurring two hops and a resulting metric of 20.

Overview

In IP networks, the bulk of traffic often goes through the core network, which reduces costs but might result in increased latency. Business traffic, however, often benefits from the ability to make path-selection decisions based on other performance metrics, such as path latency, rather than relaying on the traditional path optimization based simply on IGP metrics. Optimizing a path to reduce latency can greatly benefit applications like real-time voice and video. It can also enable high performance access to financial market data where milliseconds can translate into significant gains or losses.

You can enable IS-IS link delay in IP networks. You can achieve minimum IGP metric paths by configuring IS-IS with the appropriate link cost using the default IS-IS algorithm (0). Doing so optimizes paths to the endpoint that are based strictly on the sum of the link metrics. By using the IS-IS delay flex algorithm you can optimize paths based on their end-to-end delay.

Link delay can be dynamically measured using Two-Way Active Measurement Probes (TWAMP). The routers then flood their link delay parameters. The routers in the area store these parameters in the shared Link State Database (LSDB). Ingress nodes run an SPF algorithm against the LSDB to compute paths that are optimized on various attributes, such as link colors, IGP metric, traffic-engineering (TE) metric, or as shown in this example, link delay.

NOTE:

1. Delay of link is measured in microseconds level. In one notification it may be 1000 microseconds, and in next notification it may be 4000 microseconds. In terms of microseconds it is 400 percent change. In terms of microseconds it is a huge change but in terms of milliseconds it is not an appreciable change. Therefore, as per the requirement, you can set the optimal value otherwise every few minutes new delay will be advertised which can lead to frequent SPF computation and routes computation.

2. During periodic announcement interval (default 120 seconds) check if max Unidirectional Link Delay measured parameter crosses periodic threshold then report the same to registered modules.

   set protocols isis interface <interface_name> delay-measurement advertisement periodic threshold < 0-100>. The default threshold value is 10 percent and the value ranges from 0 to 100.

The egress router signals which flex algorithm is desired by attaching an associated color community to routes advertised through BGP. At the sending end (the local PE that has received the tagged routes advertised by the remote PE), these color communities are used to index into a color table that resolves

the remote protocol next hop (the PE's loopback address) to a flex algorithm identifier. In the context of Layer 3 VPNs a color mapping policy is used at the ingress node to select which prefixes should have their next hops resolved via the color table.

The local PE then uses its local Flex Algorithm Definition (FAD) to map the flex algorithm identifier into a set of path selection criteria, for example "use blue links and optimize on delay". The ingress PE calculates the optimal path based on the values in the LSDB, pushes the related MPLS label stack onto the packet, and sends it to the associated next hop. This results in traffic-engineered MPLS paths using IS-IS as the signaling protocol.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: Depending on the type of MPCs in your MX Series routers you might need to explicitly enable enhanced IP services to support the IS-IS delay feature. When you commit the `set chassis network-services enhanced-ip` configuration statement, you will be prompted to reboot the system.

PE1

```
set system host-name PE1
set chassis network-services enhanced-ip
set services rpm twamp server authentication-mode none
set services rpm twamp server light
set interfaces ge-0/0/0 description To_R1
set interfaces ge-0/0/0 unit 0 family inet address 10.0.1.10/24
set interfaces ge-0/0/0 unit 0 family iso
```

```
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10:0:1::10/80
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 16
set interfaces ge-0/0/1 description To_R2
set interfaces ge-0/0/1 unit 0 family inet address 10.0.2.10/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8:10:0:2::10/80
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 16
set interfaces lo0 unit 0 family inet address 192.168.255.10/32
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
set interfaces lo0 unit 0 family iso address 49.0001.000a.0a0a.0a00
set interfaces lo0 unit 0 family inet6 address 2001:db8:192:168:255::10/128
set interfaces lo0 unit 1 family inet address 172.16.10.1/32
set interfaces lo0 unit 1 family inet6 address 2001:db8:172:16:10::1/128
set interfaces lo0 unit 2 family inet address 172.16.10.2/32
set interfaces lo0 unit 2 family inet6 address 2001:db8:172:16:10::2/128
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.10/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128 index
1280
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129 index
1290
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1000
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set policy-options policy-statement prefix-sid term 2 from route-filter
2001:db8:192:168:255::10/128 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128 index
4280
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129 index
4290
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 4000
set policy-options policy-statement prefix-sid term 2 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 2 then accept
set policy-options policy-statement v6vpn1_res_map1 from route-filter 2001:db8:172:16:1::/80
orlonger
```

```
set policy-options policy-statement v6vpn1_res_map1 then accept
set policy-options policy-statement v6vpn1_res_map1 then resolution-map map1
set policy-options policy-statement v6vpn2_res_map1 from route-filter 2001:db8:172:16:2::/80
orlonger
set policy-options policy-statement v6vpn2_res_map1 then accept
set policy-options policy-statement v6vpn2_res_map1 then resolution-map map1
set policy-options policy-statement vpn1_res_map1 term 1 from route-filter 172.16.1.0/24 orlonger
set policy-options policy-statement vpn1_res_map1 term 1 then accept
set policy-options policy-statement vpn1_res_map1 term 1 then resolution-map map1
set policy-options policy-statement vpn2_res_map1 term 1 from route-filter 172.16.2.0/24 orlonger
set policy-options policy-statement vpn2_res_map1 term 1 then accept
set policy-options policy-statement vpn2_res_map1 term 1 then resolution-map map1
set policy-options resolution-map map1 mode ip-color
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 64512:1
set routing-instances vpn1 vrf-target target:64512:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn2 instance-type vrf
set routing-instances vpn2 interface lo0.2
set routing-instances vpn2 route-distinguisher 64512:2
set routing-instances vpn2 vrf-target target:64512:2
set routing-instances vpn2 vrf-table-label
set protocols bgp group to-RRv6 type internal
set protocols bgp group to-RRv6 local-address 2001:db8:192:168:255::10
set protocols bgp group to-RRv6 import v6vpn1_res_map1
set protocols bgp group to-RRv6 import v6vpn2_res_map1
set protocols bgp group to-RRv6 family inet6 unicast extended-nexthop-color
set protocols bgp group to-RRv6 family inet6-vpn unicast
set protocols bgp group to-RRv6 neighbor 2001:db8:192:168:255::2
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.10
set protocols bgp group to-RR import vpn1_res_map1
set protocols bgp group to-RR import vpn2_res_map1
set protocols bgp group to-RR family inet unicast extended-nexthop-color
set protocols bgp group to-RR family inet-vpn unicast
set protocols bgp group to-RR family traffic-engineering unicast
set protocols bgp group to-RR neighbor 192.168.255.2
set protocols bgp group to-RR vpn-apply-export
set protocols isis interface ge-0/0/0.0 level 2 post-convergence-lfa node-protection
set protocols isis interface ge-0/0/0.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 post-convergence-lfa node-protection
```

```
set protocols isis interface ge-0/0/1.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 80000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis source-packet-routing flex-algorithm 129
set protocols isis level 1 disable
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering l3-unicast-topology
set protocols isis traffic-engineering advertisement always
set protocols isis export prefix-sid
set protocols mpls traffic-engineering
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls icmp-tunneling
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group RED
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
set routing-options flex-algorithm 128 definition metric-type delay-metric
set routing-options flex-algorithm 128 definition spf
set routing-options flex-algorithm 128 definition admin-group include-any BLUE
set routing-options flex-algorithm 129 definition metric-type igp-metric
set routing-options flex-algorithm 129 definition spf
set routing-options flex-algorithm 129 definition admin-group include-any RED
set routing-options flex-algorithm 129 definition admin-group include-any BLUE
set routing-options router-id 192.168.255.10
set routing-options autonomous-system 64512
set routing-options forwarding-table export pplb
set routing-options forwarding-table ecmp-fast-reroute
set routing-options forwarding-table chained-composite-next-hop ingress l3vpn
```

P1

```
set system host-name P1
set chassis network-services enhanced-ip
set services rpm twamp server authentication-mode none
set services rpm twamp server light
set interfaces ge-0/0/0 description To_R0
set interfaces ge-0/0/0 unit 0 family inet address 10.0.1.1/24
```

```
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10:0:1::1/80
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 16
set interfaces ge-0/0/1 description To_R2
set interfaces ge-0/0/1 unit 0 family inet address 10.0.12.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8:10:0:12::1/80
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 16
set interfaces ge-0/0/2 description To_R3
set interfaces ge-0/0/2 unit 0 family inet address 10.0.13.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8:10:0:13::1/80
set interfaces ge-0/0/2 unit 0 family mpls maximum-labels 16
set interfaces lo0 unit 0 family inet address 192.168.255.1/32
set interfaces lo0 unit 0 family iso address 49.0001.0005.0505.0500
set interfaces lo0 unit 0 family inet6 address 2001:db8:192:168:255::1/128
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.1/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128 index
1281
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129 index
1291
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1001
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set policy-options policy-statement prefix-sid term 2 from route-filter
2001:db8:192:168:255::1/128 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128 index
4281
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129 index
4291
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 4001
set policy-options policy-statement prefix-sid term 2 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 2 then accept
set protocols isis interface ge-0/0/0.0 level 2 post-convergence-lfa node-protection
```

```
set protocols isis interface ge-0/0/0.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 post-convergence-lfa node-protection
set protocols isis interface ge-0/0/1.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 post-convergence-lfa node-protection
set protocols isis interface ge-0/0/2.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 80000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis source-packet-routing flex-algorithm 129
set protocols isis level 1 disable
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering l3-unicast-topology
set protocols isis traffic-engineering advertisement always
set protocols isis export prefix-sid
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls icmp-tunneling       set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group RED
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
set protocols mpls interface ge-0/0/2.0 admin-group BLUE
set routing-options router-id 192.168.255.1
set routing-options autonomous-system 65412
set routing-options forwarding-table export pplb
```

P2

```
set system host-name P2
set chassis network-services enhanced-ip
set services rpm twamp server authentication-mode none
set services rpm twamp server light
set interfaces ge-0/0/0 unit 0 family inet address 10.0.2.2/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10:0:2::2/80
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 16
set interfaces ge-0/0/1 description To_R1
set interfaces ge-0/0/1 unit 0 family inet address 10.0.12.2/24
```

```
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8:10:0:12::2/80
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 16
set interfaces ge-0/0/2 description To_R3
set interfaces ge-0/0/2 unit 0 family inet address 10.0.23.2/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8:10:0:23::2/80
set interfaces ge-0/0/2 unit 0 family mpls maximum-labels 16
set interfaces lo0 unit 0 family inet address 192.168.255.2/32
set interfaces lo0 unit 0 family iso address 49.0001.0002.0202.0200
set interfaces lo0 unit 0 family inet6 address 2001:db8:192:168:255::2/128
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.2/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128 index
1282
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129 index
1292
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1002
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set policy-options policy-statement prefix-sid term 2 from route-filter
2001:db8:192:168:255::2/128 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128 index
4282
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129 index
4292
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 4002
set policy-options policy-statement prefix-sid term 2 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 2 then accept
set policy-options policy-statement ted2nlri_igp term 1 from family traffic-engineering
set policy-options policy-statement ted2nlri_igp term 1 from protocol isis
set policy-options policy-statement ted2nlri_igp term 1 then accept
set protocols bgp group to-RRv6 type internal
set protocols bgp group to-RRv6 local-address 2001:db8:192:168:255::2
set protocols bgp group to-RRv6 family inet6 unicast
```

```
set protocols bgp group to-RRv6 family inet6-vpn unicast
set protocols bgp group to-RRv6 neighbor 2001:db8:192:168:255::10
set protocols bgp group to-RRv6 neighbor 2001:db8:192:168:255::3
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.2
set protocols bgp group to-RR family inet unicast
set protocols bgp group to-RR family inet-vpn unicast
set protocols bgp group to-RR neighbor 192.168.255.10
set protocols bgp group to-RR neighbor 192.168.255.3
set protocols bgp cluster 192.168.255.2
set protocols isis interface ge-0/0/0.0 level 2 post-convergence-lfa node-protection
set protocols isis interface ge-0/0/0.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 post-convergence-lfa node-protection
set protocols isis interface ge-0/0/1.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 post-convergence-lfa node-protection
set protocols isis interface ge-0/0/2.0 delay-metric 20000
set protocols isis interface ge-0/0/2.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 80000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis source-packet-routing flex-algorithm 129
set protocols isis level 1 disable
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering l3-unicast-topology
set protocols isis traffic-engineering advertisement always
set protocols isis export prefix-sid
set protocols mpls traffic-engineering
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls icmp-tunneling       set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group BLUE
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
set protocols mpls interface ge-0/0/2.0 admin-group BLUE
set routing-options router-id 192.168.255.2
set routing-options autonomous-system 64512
set routing-options forwarding-table export pplb
```

PE2

```
set system host-name PE2
set chassis network-services enhanced-ip
set services rpm twamp server authentication-mode none
set services rpm twamp server light
set interfaces ge-0/0/0 description To_R1
set interfaces ge-0/0/0 unit 0 family inet address 10.0.13.3/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10:0:13::3/80
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 16
set interfaces ge-0/0/1 description To_R2
set interfaces ge-0/0/1 unit 0 family inet address 10.0.23.3/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8:10:0:23::364/128
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 16
set interfaces lo0 unit 0 family inet address 192.168.255.3/32
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
set interfaces lo0 unit 0 family iso address 49.0001.0007.0707.0700
set interfaces lo0 unit 0 family inet6 address 2001:db8:192:168:255::3/128
set interfaces lo0 unit 1 family inet address 172.16.3.1/32
set interfaces lo0 unit 1 family inet6 address 2001:db8:172:16:3::1/128
set interfaces lo0 unit 2 family inet address 172.16.3.2/32
set interfaces lo0 unit 2 family inet6 address 2001:db8:172:16:3::2/128
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.3/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128 index
1283
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128 node-
segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129 index
1293
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1003
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept
set policy-options policy-statement prefix-sid term 2 from route-filter
2001:db8:192:168:255::3/128 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128 index
4283
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128 node-
```

```
segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129 index
4293
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129 node-
segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 4003
set policy-options policy-statement prefix-sid term 2 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 2 then accept
set policy-options policy-statement vpn_1_export term 1 from route-filter 172.16.1.0/24 orlonger
set policy-options policy-statement vpn_1_export term 1 then community add color128
set policy-options policy-statement vpn_1_export term 1 then next-hop 192.168.255.3
set policy-options policy-statement vpn_1_export term 1 then accept
set policy-options policy-statement vpn_1_export_v6 term 1 from route-filter
2001:db8:172:16:1::/80 orlonger
set policy-options policy-statement vpn_1_export_v6 term 1 then community add color128
set policy-options policy-statement vpn_1_export_v6 term 1 then next-hop 2001:db8:192:168:255::3
set policy-options policy-statement vpn_1_export_v6 term 1 then accept
set policy-options policy-statement vpn_1_export_v6 term 2 from route-filter
2001:db8:172:16:3::1/128 exact
set policy-options policy-statement vpn_1_export_v6 term 2 then community add color128
set policy-options policy-statement vpn_1_export_v6 term 2 then next-hop 2001:db8:192:168:255::3
set policy-options policy-statement vpn_1_export_v6 term 2 then accept
set policy-options policy-statement vpn_2_export term 1 from route-filter 172.16.2.0/24 orlonger
set policy-options policy-statement vpn_2_export term 1 then community add color129
set policy-options policy-statement vpn_2_export term 1 then next-hop 192.168.255.3
set policy-options policy-statement vpn_2_export term 1 then accept
set policy-options policy-statement vpn_2_export_v6 term 1 from route-filter
2001:db8:172:16:2::/80 orlonger
set policy-options policy-statement vpn_2_export_v6 term 1 then community add color129
set policy-options policy-statement vpn_2_export_v6 term 1 then next-hop 2001:db8:192:168:255::3
set policy-options policy-statement vpn_2_export_v6 term 1 then accept
set policy-options community color128 members color:0:128
set policy-options community color129 members color:0:129
set policy-options resolution-map map1 mode ip-color
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 routing-options rib vpn1.inet6.0 static route 2001:db8:172:16:1::/80
receive
set routing-instances vpn1 routing-options static route 172.16.1.0/24 receive
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 64512:1
set routing-instances vpn1 vrf-target target:64512:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn2 instance-type vrf
```

```
set routing-instances vpn2 routing-options rib vpn2.inet6.0 static route 2001:db8:172:16:2::/80
receive
set routing-instances vpn2 routing-options static route 172.16.2.0/24 receive
set routing-instances vpn2 interface lo0.2
set routing-instances vpn2 route-distinguisher 64512:2
set routing-instances vpn2 vrf-target target:64512:2
set routing-instances vpn2 vrf-table-label
set protocols bgp group to-RRv6 type internal
set protocols bgp group to-RRv6 local-address 2001:db8:192:168:255::3
set protocols bgp group to-RRv6 family inet6 unicast extended-nexthop-color
set protocols bgp group to-RRv6 family inet6-vpn unicast
set protocols bgp group to-RRv6 export vpn_1_export_v6
set protocols bgp group to-RRv6 export vpn_2_export_v6
set protocols bgp group to-RRv6 neighbor 2001:db8:192:168:255::2
set protocols bgp group to-RRv6 vpn-apply-export
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.3
set protocols bgp group to-RR family inet unicast extended-nexthop-color
set protocols bgp group to-RR family inet-vpn unicast
set protocols bgp group to-RR export vpn_1_export
set protocols bgp group to-RR export vpn_2_export
set protocols bgp group to-RR neighbor 192.168.255.2
set protocols bgp group to-RR vpn-apply-export
set protocols isis interface ge-0/0/0.0 level 2 post-convergence-lfa node-protection
set protocols isis interface ge-0/0/0.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/1.0 level 2 post-convergence-lfa node-protection
set protocols isis interface ge-0/0/1.0 delay-measurement advertisement periodic threshold 100
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/2.0 delay-metric 20000
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srgb start-label 80000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis source-packet-routing flex-algorithm 129
set protocols isis level 1 disable
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering l3-unicast-topology
set protocols isis traffic-engineering advertisement always
set protocols isis export prefix-sid
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
```

```
set protocols mpls icmp-tunneling       set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/0.0 admin-group BLUE
set protocols mpls interface ge-0/0/1.0 admin-group BLUEset routing-options router-id
192.168.255.3
set routing-options autonomous-system 64512
set routing-options forwarding-table export pplb
set routing-options forwarding-table ecmp-fast-reroute
set routing-options forwarding-table chained-composite-next-hop ingress l3vpn
```

Step-by-step Procedure

1. Configure the basic device settings such as hostname, IPv4, IPv6 addresses, loopback interface addresses, `enhanced-ip` mode, and enable the ISO and MPLS protocol families on all interfaces of all 4 routers.

```
user@PE1#
set system host-name PE1
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description To_R1
set interfaces ge-0/0/0 unit 0 family inet address 10.0.1.10/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10:0:1::10/80
set interfaces ge-0/0/0 unit 0 family mpls maximum-labels 16
set interfaces ge-0/0/1 description To_R2
set interfaces ge-0/0/1 unit 0 family inet address 10.0.2.10/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8:10:0:2::10/80
set interfaces ge-0/0/1 unit 0 family mpls maximum-labels 16
set interfaces lo0 unit 0 family inet address 192.168.255.10/32
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
set interfaces lo0 unit 0 family iso address 49.0001.000a.0a0a.0a00
set interfaces lo0 unit 0 family inet6 address 2001:db8:192:168:255::10/128
set interfaces lo0 unit 1 family inet address 172.16.10.1/32
set interfaces lo0 unit 1 family inet6 address 2001:db8:172:16:10::1/128
set interfaces lo0 unit 2 family inet address 172.16.10.2/32
set interfaces lo0 unit 2 family inet6 address 2001:db8:172:16:10::2/128
```

2. Configure the router-ID, autonomous system (AS) number, and apply a load balancing export policy to the forwarding table on all routers to enable load balancing of traffic.

```
user@PE1#
set routing-options router-id 192.168.255.10
set routing-options autonomous-system 64512
set routing-options forwarding-table export pplb
```

3. On PE1 and PE2, configure equal-cost multipath (ECMP) to enable fast reroute protection. Also configure chained composite next hop to allow the routers to point routes that share the same destination to a common forwarding next hop. This option improves forwarding information base (FIB) scaling.

```
user@PE1#
set routing-options forwarding-table ecmp-fast-reroute
set routing-options forwarding-table chained-composite-next-hop ingress l3vpn
```

4. Enable MPLS protocol processing on all interfaces at all routers. Also enable traffic engineering.

```
user@PE1#
set protocols mpls interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls traffic-engineering
```

5. Enable TWAMP probes on all routers. These probes support dynamic measurement of the link delay between each pair of routers.

```
user@PE1#
set services rpm twamp server authentication-mode none
set services rpm twamp server light
```

6. Configure the IS-IS protocol for point-to-point operation (TWAMP based delay measurements are not supported on multi-point links), and enable node protection mode for Topology-Independent Loop-Free Alternate (TILFA) operation on all interfaces. You also enable passive mode IS-IS on the loopback interface and disable IS-IS level 1 to use only IS-IS level 2. Enable traffic engineering with layer 3 unicast topology to download IGP topology into the TED. Configure IS-IS to support SPRING routed paths. The *prefix-sid* export policy is defined in a subsequent step. This policy is

used to have the local node advertise its loopback address with a mapping to one or more flex algorithms.

```
user@PE1#
set protocols isis level 1 disable
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 node-protection
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 node-protection
set protocols isis interface lo0.0 passive
set protocols isis backup-spf-options use-post-convergence-lfa maximum-backup-paths 8
set protocols isis backup-spf-options use-source-packet-routing
set protocols isis traffic-engineering l3-unicast-topology
set protocols isis traffic-engineering advertisement always
set protocols isis export prefix-sid
```

7. Configure dynamic IS-IS link delay-measurement using TWAMP probes on all IS-IS interfaces at all routers (except for the link between P2 and PE2, which uses a static delay value in this example).

```
user@PE1#
set protocols isis interface ge-0/0/0.0 delay-measurement advertisement periodic threshold
100
set protocols isis interface ge-0/0/1.0 delay-measurement advertisement periodic threshold
100
```

```
user@P1#
set protocols isis interface ge-0/0/0.0 delay-measurement advertisement periodic threshold
100
set protocols isis interface ge-0/0/1.0 delay-measurement advertisement periodic threshold
100
set protocols isis interface ge-0/0/2.0 delay-measurement advertisement periodic threshold
100
```

```
user@P2#
set protocols isis interface ge-0/0/0.0 delay-measurement advertisement periodic threshold
100
set protocols isis interface ge-0/0/1.0 delay-measurement advertisement periodic threshold
100
```

```
set protocols isis interface ge-0/0/2.0 delay-measurement advertisement periodic threshold
100
```

```
user@PE2#
set protocols isis interface ge-0/0/0.0 delay-measurement advertisement periodic threshold
100
set protocols isis interface ge-0/0/1.0 delay-measurement advertisement periodic threshold
100
```

8. Configure the static delay-metric on the link between P2 and PE2.

```
user@P2#
set protocols isis interface ge-0/0/2.0 delay-metric 20000
```

```
user@PE2#
set protocols isis interface ge-0/0/1.0 delay-metric 20000
```

9. Configure PE1 and PE2 to support two Layer 3 VPNs (VPN1 and VPN2).

```
user@PE1#
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 64512:1
set routing-instances vpn1 vrf-target target:64512:1
set routing-instances vpn1 vrf-table-label

set routing-instances vpn2 instance-type vrf
set routing-instances vpn2 interface lo0.2
set routing-instances vpn2 route-distinguisher 64512:2
set routing-instances vpn2 vrf-target target:64512:2
set routing-instances vpn2 vrf-table-label
```

NOTE: Note that the routing instances at PE2 are configured with IPv4 and IPv6 static routes. These routes are configured with the `receive` option to allow you to test connectivity using ping. The IS-IS delay feature operates the same if the Layer 3 VPN uses a dynamic routing protocol between the PE and an attached CE device.

We use static routes in this example to keep the topology simple to allow focus on the IS-IS delay optimization feature.

```
user@PE2#
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 routing-options rib vpn1.inet6.0 static route
2001:db8:172:16:1::/80 receive
set routing-instances vpn1 routing-options static route 172.16.1.0/24 receive
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 64512:1
set routing-instances vpn1 vrf-target target:64512:1
set routing-instances vpn1 vrf-table-label

set routing-instances vpn2 instance-type vrf
set routing-instances vpn2 routing-options rib vpn2.inet6.0 static route
2001:db8:172:16:2::/80 receive
set routing-instances vpn2 routing-options static route 172.16.2.0/24 receive
set routing-instances vpn2 interface lo0.2
set routing-instances vpn2 route-distinguisher 64512:2
set routing-instances vpn2 vrf-target target:64512:2
set routing-instances vpn2 vrf-table-label
```

10. Configure a map policy at PE1 to enable VPN route resolution for matching prefixes against the BGP color table. This allows you to evoke flex path forwarding algorithms on a per-prefix basis. The *map1* resolution policy is set to the `ip-color` resolution mode.

NOTE: In a Layer 3 VPN context a mapping policy is needed to select which prefixes are allowed to have their next hop resolved in the color table. Simply having routes with extended next hops and color communities attached does not result in the use of the color table, unless a mapping policy is used.

```
user@PE1#
set policy-options policy-statement vpn1_res_map1 term 1 from route-filter 172.16.1.0/24
orlonger
set policy-options policy-statement vpn1_res_map1 term 1 then accept
set policy-options policy-statement vpn1_res_map1 term 1 then resolution-map map1
set policy-options policy-statement vpn2_res_map1 term 1 from route-filter 172.16.2.0/24
orlonger
set policy-options policy-statement vpn2_res_map1 term 1 then accept
```

```
set policy-options policy-statement vpn2_res_map1 term 1 then resolution-map map1
set policy-options policy-statement v6vpn1_res_map1 from route-filter
2001:db8:172:16:1::/80 orlonger
set policy-options policy-statement v6vpn1_res_map1 then accept
set policy-options policy-statement v6vpn1_res_map1 then resolution-map map1
set policy-options policy-statement v6vpn2_res_map1 from route-filter
2001:db8:172:16:2::/80 orlonger
set policy-options policy-statement v6vpn2_res_map1 then accept
set policy-options policy-statement v6vpn2_res_map1 then resolution-map map1
set policy-options resolution-map map1 mode ip-color
```

11. Configure VPN route export policies at PE2 to attach the desired color communities to the VPN routes it advertises to PE1 (via the route reflector). Of significance here is how the routes from VPN1 have the color community for flex path 128 (optimize delay) attached, while the routes advertised from VPN2 have the 129 color community attached (optimize IGP metric).

```
user@PE2#
set policy-options policy-statement vpn_1_export term 1 from route-filter 172.16.1.0/24
orlonger
set policy-options policy-statement vpn_1_export term 1 then community add color128
set policy-options policy-statement vpn_1_export term 1 then next-hop 192.168.255.3
set policy-options policy-statement vpn_1_export term 1 then accept

set policy-options policy-statement vpn_2_export term 1 from route-filter 172.16.2.0/24
orlonger
set policy-options policy-statement vpn_2_export term 1 then community add color129
set policy-options policy-statement vpn_2_export term 1 then next-hop 192.168.255.3
set policy-options policy-statement vpn_2_export term 1 then accept

set policy-options policy-statement vpn_1_export_v6 term 1 from route-filter
2001:db8:172:16:1::/80 orlonger
set policy-options policy-statement vpn_1_export_v6 term 1 then community add color128
set policy-options policy-statement vpn_1_export_v6 term 1 then next-hop
2001:db8:192:168:255::3
set policy-options policy-statement vpn_1_export_v6 term 1 then accept
set policy-options policy-statement vpn_2_export_v6 term 1 from route-filter
2001:db8:172:16:2::/80 orlonger
set policy-options policy-statement vpn_2_export_v6 term 1 then community add color129
set policy-options policy-statement vpn_2_export_v6 term 1 then next-hop
2001:db8:192:168:255::3
set policy-options policy-statement vpn_2_export_v6 term 1 then accept
```

```
set policy-options community color128 members color:0:128
set policy-options community color129 members color:0:129
```

12. Configure BGP peering between the PE devices and the route reflector. Configure the unicast network layer reachability information (NLRI) to support extended color next hops on the PE devices. Enabling this option allows routes with color communities to have their next hop resolve through the color table. Without the extended next hop setting route with color communities undergoing normal next hop resolution and will not use flex algorithm paths.

13. You also enable support for IPv4 and IPv6 Layer 3 VPN unicast routes. On PE1 you apply the color mapping policies as import, so it can act on the routes received from the remote PE device.

```
user@PE1#
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.10
set protocols bgp group to-RR neighbor 192.168.255.2
set protocols bgp group to-RR family inet unicast extended-nexthop-color
set protocols bgp group to-RR family inet-vpn unicast
set protocols bgp group to-RR family traffic-engineering unicast
set protocols bgp group to-RR import vpn1_res_map1
set protocols bgp group to-RR import vpn2_res_map1
set protocols bgp group to-RRv6 type internal
set protocols bgp group to-RRv6 local-address 2001:db8:192:168:255::10
set protocols bgp group to-RRv6 neighbor 2001:db8:192:168:255::2
set protocols bgp group to-RRv6 family inet6 unicast extended-nexthop-color
set protocols bgp group to-RRv6 family inet6-vpn unicast
set protocols bgp group to-RRv6 import v6vpn1_res_map1
set protocols bgp group to-RRv6 import v6vpn2_res_map1
```

```
user@P2#
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.2
set protocols bgp group to-RR neighbor 192.168.255.10
set protocols bgp group to-RR neighbor 192.168.255.3
set protocols bgp cluster 192.168.255.2
set protocols bgp group to-RR family inet unicast
set protocols bgp group to-RR family inet-vpn unicast
```

On PE 2 you apply export policy to attach the desired color community to the VPN route advertisements sent to PE1. The `vpn-apply-export` option is needed at PE2 to allow the export policies to act on VPN routes advertised to remote PEs.

```
user@PE2#
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address 192.168.255.3
set protocols bgp group to-RR neighbor 192.168.255.2
set protocols bgp group to-RR family inet unicast extended-nexthop-color
set protocols bgp group to-RR family inet-vpn unicast
set protocols bgp group to-RR export vpn_1_export
set protocols bgp group to-RR export vpn_2_export
set protocols bgp group to-RR vpn-apply-export

set protocols bgp group to-RRv6 type internal
set protocols bgp group to-RRv6 local-address 2001:db8:192:168:255::3
set protocols bgp group to-RRv6 neighbor 2001:db8:192:168:255::2
set protocols bgp group to-RRv6 family inet6 unicast extended-nexthop-color
set protocols bgp group to-RRv6 family inet6-vpn unicast
set protocols bgp group to-RRv6 export vpn_1_export_v6
set protocols bgp group to-RRv6 export vpn_2_export_v6
set protocols bgp group to-RRv6 vpn-apply-export
```

14. Define the per-packet load balancing policy on all routers.

```
user@PE1#
set policy-options policy-statement pplb then load-balance per-packet
```

15. Configure support for segment routing with two flex algorithms (128 and 129) on all routers.

```
user@PE1#
set protocols isis source-packet-routing srgb start-label 80000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing flex-algorithm 128
set protocols isis source-packet-routing flex-algorithm 129
```

16. Configure all routers to advertise their loopback address with support for both the 128 and 129 flex algorithms. The `prefix-segment index` option sets the base label for each router's loopback

address. In this example the IPv4 base index and IPv6 base index is set to reflect the router number. As a result R0 (PE1) uses 1000 for IPv4 while R1 (P1) uses 1001.

```
user@PE1#
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.10/32
exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128
index 1280
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128
node-segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129
index 1290
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129
node-segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1000
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept

set policy-options policy-statement prefix-sid term 2 from route-filter
2001:db8:192:168:255::10/128 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128
index 4280
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128
node-segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129
index 4290
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129
node-segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 4000
set policy-options policy-statement prefix-sid term 2 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 2 then accept
```

```
user@P1#
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.1/32
exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128
index 1281
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128
node-segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129
index 1291
```

```
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129
node-segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1001
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept

set policy-options policy-statement prefix-sid term 2 from route-filter
2001:db8:192:168:255::1/128 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128
index 4281
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128
node-segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129
index 4291
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129
node-segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 4001
set policy-options policy-statement prefix-sid term 2 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 2 then accept
```

```
user@P2#
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.2/32
exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128
index 1282
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128
node-segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129
index 1292
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129
node-segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1002
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept

set policy-options policy-statement prefix-sid term 2 from route-filter
2001:db8:192:168:255::2/128 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128
index 4282
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128
node-segment
```

```
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129
index 4292
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129
node-segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 4002
set policy-options policy-statement prefix-sid term 2 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 2 then accept
```

```
user@PE2#
set policy-options policy-statement prefix-sid term 1 from route-filter 192.168.255.3/32
exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128
index 1283
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 128
node-segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129
index 1293
set policy-options policy-statement prefix-sid term 1 then prefix-segment algorithm 129
node-segment
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1003
set policy-options policy-statement prefix-sid term 1 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 1 then accept

set policy-options policy-statement prefix-sid term 2 from route-filter
2001:db8:192:168:255::3/128 exact
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128
index 4283
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 128
node-segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129
index 4293
set policy-options policy-statement prefix-sid term 2 then prefix-segment algorithm 129
node-segment
set policy-options policy-statement prefix-sid term 2 then prefix-segment index 4003
set policy-options policy-statement prefix-sid term 2 then prefix-segment node-segment
set policy-options policy-statement prefix-sid term 2 then accept
```

17. On all routers define the *RED* and *BLUE* MPLS administration groups, and assign the desired color to each interface. You also enable ICMP tunneling to allow trace route support in the context of MPLS based Layer 3 VPNs.

```
user@PE1#
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls icmp-tunneling set protocols mpls interface ge-0/0/0.0 admin-group RED
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
```

```
user@P1#
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls icmp-tunneling       set protocols mpls interface ge-0/0/0.0 admin-group
RED
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
set protocols mpls interface ge-0/0/2.0 admin-group BLUE
```

```
user@P2#
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls icmp-tunneling       set protocols mpls interface ge-0/0/0.0 admin-group
BLUE
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
set protocols mpls interface ge-0/0/2.0 admin-group BLUE
```

```
user@PE2#
set protocols mpls admin-groups RED 0
set protocols mpls admin-groups BLUE 1
set protocols mpls icmp-tunneling       set protocols mpls interface ge-0/0/0.0 admin-group
BLUE
set protocols mpls interface ge-0/0/1.0 admin-group BLUE
```

18. Configure the FADs at the ingress PE device (PE1) under the `routing-options` hierarchy. In this case you assign flex algorithm 128 to optimize the path based on the `delay-metric` and 129 to optimize on the `igp-metric`. In this example, flex algorithm 128 must take only blue color paths, while flex algorithm 129 can take either a blue or a red color path. In this example you define the FADs at PE1 only as we focus only on the forwarding path from PE1 to PE2.

To support bidirectional flex path forwarding you will need to define the desired FADs on the PE2 device. The P routers don't require a FAD definition as the FAD is only used by the ingress node when calculating a path to the egress node.

```
user@PE1#
set routing-options flex-algorithm 128 definition metric-type delay-metric
set routing-options flex-algorithm 128 definition spf
set routing-options flex-algorithm 128 definition admin-group include-any BLUE

set routing-options flex-algorithm 129 definition metric-type igp-metric
set routing-options flex-algorithm 129 definition spf
set routing-options flex-algorithm 129 definition admin-group include-any RED
set routing-options flex-algorithm 129 definition admin-group include-any BLUE
```

19. Enter `commit` to from the configuration mode.

Results

Check the results of the configuration:

```
user@PE1# show interfaces
```

```
ge-0/0/0 {
    description To_R1;
    unit 0 {
        family inet {
            address 10.0.1.10/24;
        }
        family iso;
        family inet6 {
            address 2001:db8:10:0:1::10/80;
        }
        family mpls {
            maximum-labels 16;
        }
    }
}
ge-0/0/1 {
    description To_R2;
    unit 0 {
        family inet {
```

```
            address 10.0.2.10/24;
        }
        family iso;
        family inet6 {
            address 2001:db8:10:0:2::10/80;
        }
        family mpls {
            maximum-labels 16;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.255.10/32;
            address 127.0.0.1/32;
        }
        family iso {
            address 49.0001.000a.0a0a.0a00;
        }
        family inet6 {
            address 2001:db8:192:168:255::10/128;
        }
    }
    unit 1 {
        family inet {
            address 172.16.10.1/32;
        }
        family inet6 {
            address 2001:db8:172:16:10::1/128;
        }
    }
    unit 2 {
        family inet {
            address 172.16.10.2/32;
        }
        family inet6 {
            address 2001:db8:172:16:10::2/128;
        }
    }
}
```

```
user@PE1# show policy-options
```

```
policy-statement pplb {
    then {
        load-balance per-packet;
    }
}
policy-statement prefix-sid {
    term 1 {
        from {
            route-filter 192.168.255.10/32 exact;
        }
        then {
            prefix-segment {
                algorithm 128 index 1280 node-segment;
                algorithm 129 index 1290 node-segment;
                index 1000;
                node-segment;
            }
            accept;
        }
    }
    term 2 {
        from {
            route-filter 2001:db8:192:168:255::10/128 exact;
        }
        then {
            prefix-segment {
                algorithm 128 index 4280 node-segment;
                algorithm 129 index 4290 node-segment;
                index 4000;
                node-segment;
            }
            accept;
        }
    }
}
policy-statement v6vpn1_res_map1 {
    from {
        route-filter 2001:db8:172:16:1::/80 orlonger;
    }
    then {
```

```
        accept;
        resolution-map map1;
    }
}
policy-statement v6vpn2_res_map1 {
    from {
        route-filter 2001:db8:172:16:2::/80 orlonger;
    }
    then {
        accept;
        resolution-map map1;
    }
}
policy-statement vpn1_res_map1 {
    term 1 {
        from {
            route-filter 172.16.1.0/24 orlonger;
        }
        then {
            accept;
            resolution-map map1;
        }
    }
}
policy-statement vpn2_res_map1 {
    term 1 {
        from {
            route-filter 172.16.2.0/24 orlonger;
        }
        then {
            accept;
            resolution-map map1;
        }
    }
}
resolution-map map1 {
    mode ip-color;
}
```

user@PE1# show protocols

```
bgp {
    group to-RRv6 {
        type internal;
        local-address 2001:db8:192:168:255::10;
        import [ v6vpn1_res_map1 v6vpn2_res_map1 ];
        family inet6 {
            unicast {
                extended-nexthop-color;
            }
        }
        family inet6-vpn {
            unicast;
        }
        neighbor 2001:db8:192:168:255::2;
    }
    group to-RR {
        type internal;
        local-address 192.168.255.10;
        import [ vpn1_res_map1 vpn2_res_map1 ];
        family inet {
            unicast {
                extended-nexthop-color;
            }
        }
        family inet-vpn {
            unicast;
        }
        family traffic-engineering {
            unicast;
        }
        neighbor 192.168.255.2;
    }
}
isis {
    interface ge-0/0/0.0 {
        level 2 {
            post-convergence-lfa {
                node-protection;
            }
        }
```

```
        delay-measurement {
            advertisement {
                periodic {
                    threshold 100;
                }
            }
        }
        point-to-point;
    }
    interface ge-0/0/1.0 {
        level 2 {
            post-convergence-lfa {
                node-protection;
            }
        }
        delay-measurement {
            advertisement {
                periodic {
                    threshold 100;
                }
            }
        }
        point-to-point;
    }
    interface lo0.0 {
        passive;
    }
    source-packet-routing {
        srgb start-label 80000 index-range 5000;
        flex-algorithm [ 128 129 ];
    }
    level 1 disable;
    backup-spf-options {
        use-post-convergence-lfa maximum-backup-paths 8;
        use-source-packet-routing;
    }
    traffic-engineering {
        l3-unicast-topology;
        advertisement always;
    }
    export prefix-sid;
}
mpls {
```

```
    traffic-engineering;
    admin-groups {
        RED 0;
        BLUE 1;
    }
    icmp-tunneling;
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface ge-0/0/0.0 {
        admin-group RED;
    }
    interface ge-0/0/1.0 {
        admin-group BLUE;
    }
}
```

user@PE1# show routing-options

```
flex-algorithm 128 {
    definition {
        metric-type delay-metric;
        spf;
        admin-group include-any BLUE;
    }
}
flex-algorithm 129 {
    definition {
        metric-type igp-metric;
        spf;
        admin-group include-any [ RED BLUE ];
    }
}
router-id 192.168.255.10;
autonomous-system 64512;
forwarding-table {
    export pplb;
    ecmp-fast-reroute;
    chained-composite-next-hop {
        ingress {
```

```
            l3vpn;
        }
    }
}
```

user@PE1# show routing-instances

```
vpn1 {
    instance-type vrf;
    interface lo0.1;
    route-distinguisher 64512:1;
    vrf-target target:64512:1;
    vrf-table-label;
}
vpn2 {
    instance-type vrf;
    interface lo0.2;
    route-distinguisher 64512:2;
    vrf-target target:64512:2;
    vrf-table-label;
}
```

user@PE1# show services rpm

```
twamp {
    server {
        authentication-mode none;
        light;
    }
}
```

Verification

IN THIS SECTION

Verify IS-IS Adjacencies

IN THIS SECTION

Purpose

Verify expected IS-IS adjacencies on the routing devices.

Action

From operational mode, enter the `show isis adjacency` command.

```
user@PE1> show isis adjacency
```

```
Interface             System         L State       Hold (secs) SNPA
ge-0/0/0.0            P1           2  Up                   26
ge-0/0/1.0            P2           2  Up                   25
```

Meaning

Th output indicates that PE1 has successfully formed IS-IS adjacencies on its `ge-0/0/0.0` and `ge-0/0/1.0` interfaces, which attach to their P1 and P2 routers, respectively.

Verify IS-IS Database

IN THIS SECTION

Purpose

Verify that link delay parameters are present in the IS-IS database.

Action

Use the `show isis database extensive | match delay` operational command.

```
user@PE1> show isis database extensive | match delay
```

```
Unidirectional link delay: 1041
        Min unidirectional link delay: 841
        Max unidirectional link delay: 1885
        Unidirectional delay variation: 71
        Unidirectional link delay: 2469
        Min unidirectional link delay: 766
        Max unidirectional link delay: 15458
        Unidirectional delay variation: 129
        Unidirectional link delay: 20000
        Min unidirectional link delay: 20000
        Max unidirectional link delay: 20000
        Unidirectional delay variation: 20000
        Unidirectional link delay: 1272
        Min unidirectional link delay: 628
        Max unidirectional link delay: 3591
        Unidirectional delay variation: 1559
        Unidirectional link delay: 8470
        Min unidirectional link delay: 855
        Max unidirectional link delay: 52934
        Unidirectional delay variation: 7900
```

```
        Unidirectional link delay: 5736
        Min unidirectional link delay: 3650
        Max unidirectional link delay: 7946
        Unidirectional delay variation: 4416
        Unidirectional link delay: 2312
        Min unidirectional link delay: 740
        Max unidirectional link delay: 14227
        Unidirectional delay variation: 3144
        Unidirectional link delay: 1233
        Min unidirectional link delay: 711
        Max unidirectional link delay: 2833
        Unidirectional delay variation: 366
        Unidirectional link delay: 928
        Min unidirectional link delay: 844
        Max unidirectional link delay: 1042
        Unidirectional delay variation: 143
        Unidirectional link delay: 7570
        Min unidirectional link delay: 761
        Max unidirectional link delay: 61926
        Unidirectional delay variation: 27290
```

Meaning

The output displays the dynamic delay that is associated with the various interfaces in the topology. The highlighted portion of the output specifies the static delay of 20000 microseconds that is configured on the P2 to PE2 link. The statically configured delay value is significantly higher than any of the dynamic delay measurements. This large delay is configured to make it easy to predict the delay optimized blue path through the network.

Verify BGP Peering

IN THIS SECTION

Purpose

Verify that both PEs have successfully established IPv4 and IPv6 peering sessions to the route reflector.

Action

Use the `show bgp summary` operational command. In this case we run the command on P2, the route reflector, as it provides a convenient location to confirm both peering sessions from both PEs using a single command.

```
user@P2 show bgp summary
```

```
Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 2 Peers: 4 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet6.0
                       0          0          0          0          0          0
bgp.l3vpn-inet6.0
                       6          6          0          0          0          0
inet.0
                       0          0          0          0          0          0
bgp.l3vpn.0
                       6          6          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.168.255.3         64512       2511       2489       0       0    18:49:42 Establ
  inet.0: 0/0/0/0
  bgp.l3vpn.0: 4/4/4/0
192.168.255.10        64512       2511       2491       0       0    18:49:46 Establ
  inet.0: 0/0/0/0
  bgp.l3vpn.0: 2/2/2/0
2001:db8:192:168:255::3       64512       2512       2490       0       0    18:49:46 Establ
  inet6.0: 0/0/0/0
  bgp.l3vpn-inet6.0: 4/4/4/0
2001:db8:192:168:255::10       64512       2510       2490       0       0    18:49:42 Establ
  inet6.0: 0/0/0/0
  bgp.l3vpn-inet6.0: 2/2/2/0
```

Meaning

The output confirms that all BGP peering sessions are established correctly. The display also confirms that Layer 3 VPN routes are being advertised/learned over these peering sessions.

Verify Color Community on VPN Routes

IN THIS SECTION

Purpose

Verify the VPN routes advertised by PE2 are correctly tagged with a color community.

Action

Use the `show route detail <prefix> table <table-name>` operational command at PE1 to display details about a Layer 3 VPN route learned from PE2.

```
user@PE1 show route detail 172.16.1.0 table vpn1
```

```
vpn1.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 64512:1
                Next hop type: Indirect, Next hop index: 0
                Address: 0xc5b9d5c
                Next-hop reference count: 3
                Source: 192.168.255.2
                Next hop type: Router, Next hop index: 0
                Next hop: 10.0.2.2 via ge-0/0/1.0 weight 0x1, selected
                Label operation: Push 81282
                Label TTL action: prop-ttl
                Load balance label: Label 81282: None;
                Label element ptr: 0xcbf1440
                Label parent element ptr: 0x0
                Label element references: 2
                Label element child references: 0
                Label element lsp id: 0
                Session Id: 0x0
```

```
Protocol next hop: 192.168.255.3-128<c>
Label operation: Push 16
Label TTL action: prop-ttl
Load balance label: Label 16: None;
Composite next hop: 0xbd50440 665 INH Session ID: 0x0
Indirect next hop: 0xc74e684 1048588 INH Session ID: 0x0
State: <Secondary Active Int Ext ProtectionCand>
Local AS: 64512 Peer AS: 64512
Age: 19:10:35   Metric2: 2204
Validation State: unverified
ORR Generation-ID: 0
Task: BGP_64512.192.168.255.2
Announcement bits (1): 0-KRT
AS path: I  (Originator)
Cluster list:  192.168.255.2
Originator ID: 192.168.255.3
Communities: target:64512:1 color:0:128
Import Accepted
VPN Label: 16
Localpref: 100
Router ID: 192.168.255.2
Primary Routing Table: bgp.l3vpn.0
Thread: junos-main
```

Meaning

The output confirms that a VPN prefix in the VPN1 routing instance has a color community `color:0:128` attached. In addition, you can confirm that the protocol next hop for this route is the loopback address of the PE2 router with an extended next hop that indexes a matching entry in the color table.

Though not show, you can repeat this command for a prefix in the VPN2 table. You expect to find these routes have the `color:0:129` attached.

Verify junos-rti-tc-<color>.inet.0 Routing Table

IN THIS SECTION

Purpose

Verify the `junos-rti-tc-<color>.inet.0` routing table is correctly populated with all router IDs (loopback addresses) showing support for both the 128 and 129 flex algorithms.

NOTE: IPv6 routes are supported via the `junos-rti-tc-<color>.inet6.0` table. You can verify this table using the same approach as shown in this section for the IPv4 color table.

Action

Use the `show route table junos-rti-tc-<color>.inet.0` operational command.

```
user@PE1> show route table junos-rti-tc-<color>.inet.0
```

```
junos-rti-tc-<color>.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.255.1-128<c>/64
                   *[L-ISIS/14] 6d 14:40:37, metric 1527
                    >  to 10.0.2.2 via ge-0/0/1.0, Push 81281
192.168.255.1-129<c>/64
                   *[L-ISIS/14] 6d 14:40:35, metric 10
                    >  to 10.0.1.1 via ge-0/0/0.0
                       to 10.0.2.2 via ge-0/0/1.0, Push 81291
192.168.255.2-128<c>/64
                   *[L-ISIS/14] 6d 14:40:40, metric 761
                    >  to 10.0.2.2 via ge-0/0/1.0
192.168.255.2-129<c>/64
                   *[L-ISIS/14] 6d 14:40:35, metric 10
                    >  to 10.0.2.2 via ge-0/0/1.0
                       to 10.0.1.1 via ge-0/0/0.0, Push 81292
192.168.255.3-128<c>/64
                   *[L-ISIS/14] 6d 14:40:37, metric 2382
                    >  to 10.0.2.2 via ge-0/0/1.0, Push 81283



                  192.168.255.3-129<c>/64
                   *[L-ISIS/14] 6d 14:40:35, metric 20
                    >  to 10.0.1.1 via ge-0/0/0.0, Push 81293
```

```
                    to 10.0.2.2 via ge-0/0/1.0, Push 81293
```

Meaning

The output displays the routes in the `junos-rti-tc-<color>.inet.0` route table. The highlighted portion indicates the two routes originate from PE2. The `192.168.255.3-128<c>` route has only one possible path and takes the `ge-0/0/1.0` interface to P2 as a next hop. Recall that the 128 flex algorithm must use blue links, and from the perspective of PE1, which leaves only the blue colored `ge-0/0/1` interface as a viable path.

In contrast, the route for `192.168.255.3-129<c>` is able to load balance over both the `ge-0/0/0.0` interfaces to P1 and the `ge-0/0/1.0` to P2. Recall that this path for flex algorithm can take any path that is either blue or red, thus can use either of its interfaces when forwarding to its associated destination.

Verify TWAMP Operation

IN THIS SECTION

Purpose

Verify that TWAMP probes are operating between routers with dynamic link delay configured.

Action

Use the `show services rpm twamp client` operational mode command.

```
user@PE1> show services rpm twamp client
```

```
Connection      Session         Sender          Sender Reflector       Reflector
Name            Name            address         port   address         port
```

```
__r__8          __r__9          10.0.1.10         56570 10.0.1.1          862
__r__10         __r__11         10.0.2.10         64074 10.0.2.2          862
```

Meaning

The highlighted portion of the output indicates that PE1 has two TWAMP neighbors: P2 (10.0.1.2) and P1 (10.0.1.1).

If desired use the `show services rpm twamp client probe-results` operational mode command to see the current and historical delay measurement values.

```
user@PE1> show services rpm twamp client probe-results
```

```
root@PE1# run show services rpm twamp client probe-results
    Owner: __r__12, Test: __r__13
    TWAMP-Server-Status: Light, Number-Of-Retries-With-TWAMP-Server: 0
    Reflector address: 10.0.2.2, Reflector port: 862, Sender address: 10.0.2.10, sender-port:
57270
    Test size: 10 probes
    Probe results:
      Response received
      Probe sent time: Thu May  6 14:43:26 2021
      Probe rcvd/timeout time: Thu May  6 14:43:26 2021
      Rtt: 1931 usec, Egress jitter: 259 usec, Ingress jitter: 96 usec, Round trip jitter: 353
usec
      Egress interarrival jitter: 5489 usec, Ingress interarrival jitter: 855 usec, Round trip
interarrival jitter: 6076 usec
    Results over current test:
      Probes sent: 8, Probes received: 8, Loss percentage: 0.000000
      Measurement: Round trip time
        Samples: 8, Minimum: 1576 usec, Maximum: 13289 usec, Average: 6100 usec, Peak to peak:
11713 usec, Stddev: 4328 usec,
        Sum: 48797 usec
      Measurement: Ingress delay
        Samples: 2, Minimum: 8466 usec, Maximum: 8488 usec, Average: 8477 usec, Peak to peak: 22
usec, Stddev: 11 usec,
        Sum: 16954 usec
      Measurement: Egress delay
        Samples: 2, Minimum: 118 usec, Maximum: 4801 usec, Average: 2460 usec, Peak to peak:
4683 usec, Stddev: 2342 usec,
```

```
        Sum: 4919 usec
      Measurement: Positive egress jitter
        Samples: 4, Minimum: 259 usec, Maximum: 11250 usec, Average: 4465 usec, Peak to peak:
10991 usec, Stddev: 4225 usec,
        Sum: 17859 usec
      Measurement: Negative egress jitter
        Samples: 4, Minimum: 201 usec, Maximum: 6564 usec, Average: 4467 usec, Peak to peak:
6363 usec, Stddev: 2566 usec,
        Sum: 17869 usec
      Measurement: Positive ingress jitter
        Samples: 5, Minimum: 96 usec, Maximum: 4954 usec, Average: 1431 usec, Peak to peak: 4858
usec, Stddev: 1843 usec,
        Sum: 7155 usec
      Measurement: Negative ingress jitter
        Samples: 3, Minimum: 202 usec, Maximum: 4990 usec, Average: 2340 usec, Peak to peak:
4788 usec, Stddev: 1988 usec,
        Sum: 7021 usec
      Measurement: Positive round trip jitter
        Samples: 4, Minimum: 353 usec, Maximum: 11585 usec, Average: 5827 usec, Peak to peak:
11232 usec, Stddev: 4797 usec,
        Sum: 23309 usec
      Measurement: Negative round trip jitter
        Samples: 4, Minimum: 2056 usec, Maximum: 9734 usec, Average: 5831 usec, Peak to peak:
7678 usec, Stddev: 2776 usec,
        Sum: 23325 usec
    Results over last test:
. . .
```

Verify Route Resolution

IN THIS SECTION

Purpose

Verify the routes for the VPN1 and VPN2 resolve over the expected flex algorithm paths.

Action

Use the `show route` operational mode command.

```
user@PE1> show route 172.16.1.0
```

```
inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both
. . .
vpn1.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 6d 16:32:32, localpref 100, from 192.168.255.2
                      AS path: I, validation-state: unverified
                    >  to 10.0.2.2 via ge-0/0/1.0, Push 16, Push 81287(top)
```

```
user@PE1> show route 172.16.2.0
```

```
inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both. . .

vpn2.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.2.0/24      *[BGP/170] 6d 16:36:02, localpref 100, from 192.168.255.2
                      AS path: I, validation-state: unverified
                       to 10.0.1.1 via ge-0/0/0.0, Push 17, Push 81297(top)
                    >  to 10.0.2.2 via ge-0/0/1.0, Push 17, Push 81297(top)
```

Meaning

The highlighted output indicates that on the PE1 device, the 172.16.1.0 route for VPN1 uses FAD 128 taking only the blue color path, which makes P1 (10.0.2.2) its next hop while the route for VPN2, 172.16.2.0 uses FAD 129, which means it can take the red color path either through ge-0/0/0.0 interface to P1>PE2 or through the ge-0/0/1.0 interface to P2> PE2. This is also true for IPv6 routes, as shown here for VPN1:

```
user@PE1> show route 2001:db8:172:16:1::/80
```

```
vpn1.inet6.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8:172:16:1::/80
                   *[BGP/170] 01:26:27, localpref 100, from 2001:db8:192:168:255::2
                      AS path: I, validation-state: unverified
                    >  to fe80::5668:a5ff:fed1:21d9 via ge-0/0/1.0, Push 16, Push 84287(top)
```

The IPv6 route from VPN1 resolves to the same forwarding path as its IPv4 counterpart, which makes sense as they are both using flex algorithm 128 to force the use of blue links with delay optimization. Recall that you configured PE2, the source of these routes, to use a label base of 1287 for IPv4 routes and 4287 for IPv6 routes, and that the `source-packet-routing srgb start-label` to 8000. As a result the IPv4 route from VPN1 has a label of 81287 while the IPv6 route from VPN1 uses 84287.

Verify Forwarding Paths

IN THIS SECTION

Purpose

Verify the routes for VPN1 and VPN2 are forwarded over the expected flex algorithm paths.

Action

Use the `ping` and `trace route` operational mode commands to verify reachability, and to confirm the IPv4 forwarding path used by PE1 when sending traffic to VPN destinations as PE2.

NOTE: The use of static routes with a receive next hop at PE2 allows you to ping the remote routes. You can expect the last hop of the trace route to timeout, however, as trace route processing is not supported when targeting an IPv4 static receive route.

```
user@PE1> ping 172.16.1.0 routing-instance vpn1 count 2
```

```
PING 172.16.1.0 (172.16.1.0): 56 data bytes
64 bytes from 172.16.1.0: icmp_seq=0 ttl=63 time=6.617 ms
64 bytes from 172.16.1.0: icmp_seq=1 ttl=63 time=33.849 ms

--- 172.16.1.0 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 6.617/20.233/33.849/13.616 ms
```

```
user@PE1> traceroute 172.16.1.0 routing-instance vpn1 no-resolve
```

```
traceroute to 172.16.1.0 (172.16.1.0), 30 hops max, 52 byte packets
 1  10.0.2.2 (10.0.2.2)  4.729 ms  4.698 ms  4.559 ms
     MPLS Label=81282 CoS=0 TTL=1 S=0
     MPLS Label=16 CoS=0 TTL=1 S=1
 2  10.0.12.1 (10.0.12.1)  8.524 ms  7.780 ms  4.338 ms
     MPLS Label=81282 CoS=0 TTL=1 S=0
     MPLS Label=16 CoS=0 TTL=2 S=1
 3  * * *
```

```
*^C
user@PE1>
```

```
user@PE1> ping 172.16.2.0 routing-instance vpn1 count 2
```

```
PING 172.16.2.0 (172.16.2.0): 56 data bytes
64 bytes from 172.16.2.0: icmp_seq=0 ttl=63 time=31.723 ms
64 bytes from 172.16.2.0: icmp_seq=1 ttl=63 time=3.873 ms

--- 172.16.2.0 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 3.873/17.798/31.723/13.925 ms
```

```
user@PE1> traceroute 172.16.2.0 routing-instance vpn2 no-resolve
```

```
traceroute to 172.16.2.0 (172.16.2.0), 30 hops max, 52 byte packets
 1  10.0.1.1  7.102 ms  8.746 ms  7.820 ms
     MPLS Label=81292 CoS=0 TTL=1 S=0
     MPLS Label=17 CoS=0 TTL=1 S=1
 2  * * *
*^C
user@PE1>
```

Meaning

The output indicates that the expected forwarding paths are used. For example, the trace route for the 172.16.1.0/24 route in VPN1 shows that blue paths are used, and that the high-delay link between P2 and PE2 is avoided. This confirms that flex algorithm prefers a path with an extra hop if it results in a reduction of end-to-end path latency. In this case the 10.0.12.0 link between P2 and P1 is used while the direct link between P2 and PE2 is avoided.

In contrast, the path taken for the 172.16.2.0/24 route, associated with VPN2 and flex algorithm 129, is able to take either of the direct paths between PE1 and PE2. In this case the forwarding path is from PE1 to P1 and then to the destination (PE2), where as noted the last hop times out. This timeout on the last hop does not occur for routes that point to a CE device (as opposed to the static receive routes used in this example).

Though not show here for brevity, you expect the same forwarding paths for trace routes to the IPv6 VPN routes based on whether they are mapped to flex algorithm 128 or 129, which in this example means associated with VPN1 versus VPN2, respectively.

Configuring OSPF Link Delay and Delay Normalization on an OSPF Interface

In IP networks, the bulk of traffic often goes through the core network, which reduces costs but might result in increased latency. Business traffic, however, often benefits from the ability to make path-selection decisions based on other performance metrics, such as path latency, rather than relaying on the traditional path optimization based simply on IGP metrics. Optimizing a path to reduce latency can greatly benefit applications like real-time voice and video. It can also enable high performance access to financial market data where milliseconds can translate into significant gains or losses.

You can enable OSPF link delay in IP networks. You can achieve minimum IGP metric paths by configuring OSPF with the appropriate link cost using the default OSPF algorithm. Doing so optimizes paths to the endpoint that are based strictly on the sum of the link metrics. By using the OSPF delay flex algorithm you can optimize paths based on their end-to-end delay.

Link delay can be dynamically measured using Two-Way Active Measurement Probes (TWAMP). The routers then flood their link delay parameters. The routers in the area store these parameters in the shared Link State Database (LSDB). Ingress nodes run an SPF algorithm against the LSDB to compute paths that are optimized on various attributes, such as link colors, IGP metric, traffic-engineering (TE) metric.

To configure link delay measurement for an OSPF Interface:

1. Create an OSPF area.

```
[edit protocols]
user@host#set protocols ospf area area-id
```

   For example:

```
[edit protocols]
user@host#set protocols ospf area 0.0.0.0
```

2. Specify the interface.

```
[edit protocols ospf area 0.0.0.0]
user@host#set interface interface-name
```

For example:

```
[edit protocols ospf area 0.0.0.0]
user@host#set interface ge-0/0/0.0
```

3. Configure dynamic OSPF link delay-measurement on the OSPF interface of the device.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement
```

4. Configure the delay-measurement advertisement on the OSPF interface of the device. You can either configure accelerated or periodic advertisement.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement advertisement (accelerated | periodic)
```

NOTE: Accelerated advertisement is disabled by default. To configure accelerated advertisement, configure the threshold percentage.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement advertisement accelerated threshold percentage
```

5. To configure periodic advertisement, you can either configure interval or the threshold percentage.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement advertisement periodic (interval interval  seconds |
threshold threshold percentage)
```

For example: To configure periodic interval:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement advertisement periodic interval 35
```

For example: To configure periodic threshold:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement advertisement periodic threshold 100)
```

6. (Optional) Specify the probe count.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement probe-count seconds
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement probe-count 10
```

7. (Optional) Specify the probe interval.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement probe-interval seconds
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement probe-interval 100
```

8. (Optional) Specify the normalize interval and offset values.

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement normalize interval microseconds offset microseconds seconds
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-measurement normalize interval 50 offset 10
```

9. Enter `commit` from the configuration mode.

To configure delay metric for an OSPF Interface:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-metric microseconds
```

For example:

```
[edit protocols ospf area 0.0.0.0 interface ge-0/0/0.0]
user@host#set delay-metric 20000
```

Enter `commit` from the configuration mode.

To verify your configuration results, use the `show protocols` operational command.

`user@host# show protocols`

```
ospf {
          area 0.0.0.0
             interface ge-0/0/0.0 {
                    delay-measurement {
    advertisement {
       accelerated {
          threshold 100;
}
       periodic {
          interval 35;
          threshold 100;
    }
  }
     probe-count 10;
     probe-interval 100;
}
                delay-metric {
                  20000;
    )
}
     normalize interval 50 offset 10;
```

To verify that link delay parameters are present in the OSPF database use the `show ospf database extensive | match delay` operational command.

```
user@host> show ospf database extensive | match delay
```

```
        Unidirectional link delay: 20000
        Min unidirectional link delay: 20000
        Max unidirectional link delay: 20000
        Unidirectional delay variation: 20000
```

The output displays the delay of 20000 microseconds that is configured on the interface.

To verify the normalized delay value, use the `show ospf interface extensive` operational command.

```
user@host> show ospf interface extensive
```

```
root@R0# run show ospf interface ge-0/0/0.0 extensive
Interface           State   Area            DR ID           BDR ID          Nbrs
ge-0/0/0.0          PtToPt  0.0.0.0         0.0.0.0         0.0.0.0            1
  Type: P2P, Address: 21.0.1.1, Mask: 255.255.255.0, MTU: 1500, Cost: 1
  Adj count: 1
  Hello: 10, Dead: 40, ReXmit: 5, Not Stub
  Auth type: None
  Protection type: None
  Topology default (ID 0) -> Cost: 1
  Unidirectional link delay: 1616
  Min unidirectional link delay: 712
  Max unidirectional link delay: 8125
  Unidirectional delay variation: 2992
  Delay Normalization: Enabled (Interval: 50, Offset: 10)
  Normalized Delay: 860 (Last Normalized: Mon Jul 24 10:03:41)
```

The output displays that the unidirectional link delay 1616 microseconds has been normalized into a value (860) that OSPF uses in SPF computations for flexible algorithm. Last Normalized: Mon Jul 24 10:03:41 indicates the timestamp of the most recent normalization calculation.

SEE ALSO

delay-measurement (Protocols OSPF)

delay-metric (Protocols OSPF)

Understand Two-Way Active Measurement Protocol

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
25.4R1	Starting in Junos OS and Junos OS Evolved Release 25.4R1, you can use delay normalization for OSPFv2 to compute and advertise a normalized delay metric for flexible algorithm, to improve path-selection consistency across all IGP instances.
21.4R1	Starting in Junos OS Release 21.4R1, you can get the measurement of various performance metrics in IP networks, by using probe messages.

Color-Based Traffic Engineering Configuration

SUMMARY

IN THIS SECTION

BGP Classful Transport Planes Overview

IN THIS SECTION

Benefits of BGP Classful Transport Planes

- **Network-slicing**–Service and transport layers are decoupled from each other, laying the foundation for network-slicing and virtualization with the end-to-end slicing across multiple domains, thereby significantly reducing the CAPEX.
- **Inter-domain interoperability**–Extends transport class deployment across co-operating domains so the different transport signaling protocols in each domain interoperate. Reconciles any differences between extended community namespaces that may be in use in each domain.
- **Colored resolution with fallback**–Enables resolution over colored tunnels (RSVP, IS-IS flexible algorithm) with flexible fallback options over best-effort tunnels or any other color tunnel.
- **Quality-of-service**–Customizes and optimizes the network to achieve the end-to-end SLA requirements.
- **Leveraging existing deployments**–Supports well deployed tunneling protocols like RSVP along with new protocols, such as IS-IS flexible algorithm, preserving ROI and reducing OPEX.

Terminology of BGP Classful Transport Planes

This section provides a summary of commonly used terms for understanding BGP classful transport plane.

- **Service node**–Ingress Provider Edge (PE) devices that send and receive service routes (Internet and Layer 3 VPN).
- **Border node**–Device at the connection point of different domains (IGP areas or ASs).
- **Transport node**–Device that sends and receives BGP-Labeled Unicast (LU) routes.
- **BGP-VPN**–VPNs built using RFC4364 mechanisms.
- **Route Target (RT)**–Type of extended community used to define VPN membership.
- **Route Distinguisher (RD)**–Identifier used to distinguish to which VPN or virtual private LAN service (VPLS) a route belongs. Each routing instance must have a unique route distinguisher associated with it.

- **Resolution scheme**–Used to resolve protocol next-hop address (PNH) in resolution RIBs providing fallback.

  They map the routes to the different transport RIBs in the system based on mapping community.

- **Service family**–BGP address family used for advertising routes for data traffic, as opposed to tunnels.
- **Transport family** –BGP address family used for advertising tunnels, which are in turn used by service routes for resolution.
- **Transport tunnel**–A tunnel over which a service may place traffic, for example, GRE, UDP, LDP, RSVP, SR-TE, BGP-LU.
- **Tunnel domain**–A domain of the network containing service nodes and border nodes under a single administrative control that has a tunnel between them. An end-to-end tunnel spanning several adjacent tunnel domains can be created by stitching the nodes together using labels.
- **Transport class**–A group of transport tunnels offering the same type of service.
- **Transport class RT**–A new format of route target used to identify a specific transport class.

  A new format of Route Target used to identify a specific transport class.

- **Transport RIB**–At the service node and border node, a transport class has an associated transport RIB that holds its tunnel routes.
- **Transport RTI**–A routing instance; container of transport RIB, and associated transport class Route Target and Route Distinguisher.
- **Transport plane**–Set of transport RTIs importing same transport class RT. These are in turn stitched together to span across tunnel domain boundaries using a mechanism similar to Inter-AS option-b to swap labels at border nodes (nexthop-self), forming an end-to-end transport plane.
- **Mapping community**–Community on a service route that maps to resolve over a transport class.

Understanding BGP Classful Transport Planes

You can use BGP classful transport planes to configure transport classes for classifying a set of transport tunnels in an intra-AS network based on the traffic engineering characteristics and use these transport tunnels to map service routes with the desired SLA and intended fallback.

BGP classful transport planes can extend these tunnels to inter-domain networks that span across multiple domains (ASs or IGP areas) while preserving the transport class. To do this, you must configure the BGP classful transport transport layer BGP family between the border and service nodes.

In both inter-AS and intra-AS implementations, there can be many transport tunnels (MPLS LSPs, IS-IS flexible algorithm, SR-TE) created from the service and border nodes. The LSPs may be signaled using different signaling protocols in different domains, and can be configured with different traffic

engineering characteristics (class or color). The transport tunnel endpoint also acts as the service endpoint and can be present in the same tunnel domain as the service ingress node, or in an adjacent or non-adjacent domain. You can use BGP classful transport planes to resolve services over LSPs with certain traffic engineering characteristics either inside a single domain or across multiple domains.

BGP classful transport planes reuse the BGP-VPN technology, keeping the tunneling-domains loosely coupled and coordinated.

- The network layer reachability information (NLRI) is **RD:TunnelEndpoint** used for path-hiding.
- The route target indicates the transport class of the LSPs, and leaks routes to the corresponding transport RIB on the destination device.
- Every transport tunneling protocol installs an ingress route into the transport-class.inet.3 routing table, models the tunnel transport class as a VPN route target, and collects the LSPs of the same transport class in the transport-class.inet.3 transport-rib routing table.
- Routes in this routing instance are advertised in the BGP classful transport plane (inet transport) AFI-SAFI following procedures similar to RFC-4364.
- When crossing inter-AS link boundary, you must follow Option-b procedures to stitch the transport tunnels in these adjacent domains.

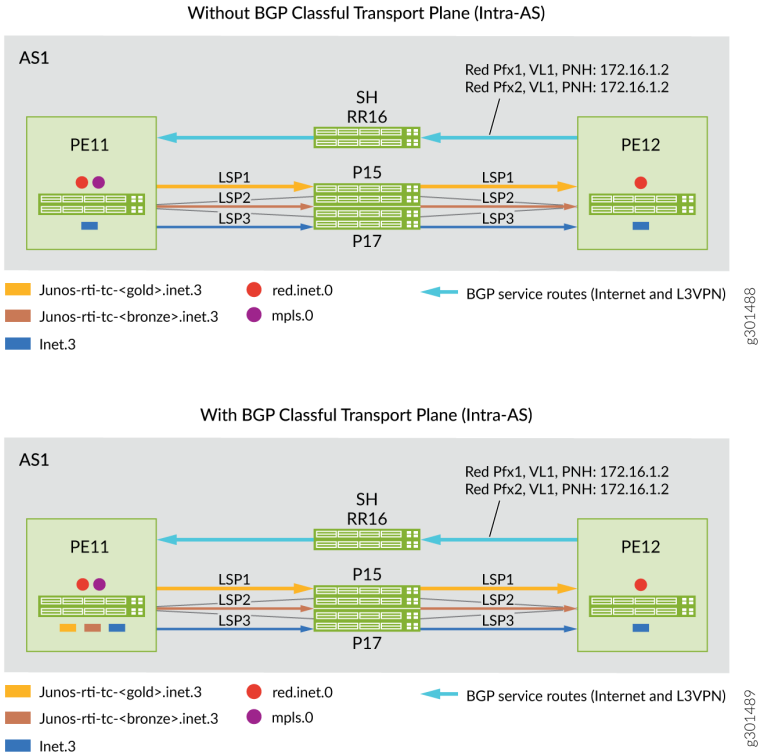
  Similarly, when crossing intra-AS regions you must follow Option-b procedures to stitch the transport tunnels in the different TE-domains.
- You can define resolution schemes to specify the intent on the variety of transport classes in a fallback order.
- You can resolve service routes and BGP classful transport routes over these transport classes, by carrrying the mapping community on them.

The BGP classful transport family runs along side the BGP-LU transport layer family. In a seamless MPLS network running BGP-LU, meeting stringent SLA requirements of 5G is a challenge as the traffic engineering characteristics of the tunnels are not known or preserved across domain boundaries. BGP classful transport planes provide operationally easy and scalable means to advertise multiple paths for remote loopbacks along with the transport class information in the seamless MPLS architecture. In BGP classful transport family routes, different SLA paths are represented using Transport Route-Target extended community, which carries the transport class color. This Transport Route-Target is used by the receiving BGP routers to associate the BGP classful transport route with the appropriate transport class. When re-advertising the BGP classful transport routes, MPLS swaps routes, inter connect the intra-AS tunnels of the same transport class, thereby forming an end-to-end tunnel that preserves the transport class.

Intra-AS Implementation of BGP Classful Transport Planes

Figure 24 on page 284 illustrates a network topology with before-and-after scenarios of implementing BGP classful transport planes in an intra-AS domain. Devices PE11 and PE12 use RSVP LSPs as the transport tunnel and all transport tunnel routes are installed in inet.3 RIB. Implementing BGP classful transort planes enables RSVP transport tunnels to be color-aware similar to segment routing tunnels.

Figure 24: Intra-AS Domain: Before-and-After Scenarios For BGP Classful Transport Planes Implementation



To classify transport tunnels into BGP transport class in an intra-AS setup:

1. Define the transport class at the service node (ingress PE devices), for example, gold and bronze, and assign color community values to the defined transport class.

   Sample configuration:

```
pe11# show routing-options
route-distinguisher-id 172.16.1.1;
transport-class {
    name gold {
        color 100;
```

```
    }
    name bronze {
        color 200;
```

2. Associate the transport tunnel to a specific transport class at the ingress node of the tunnels.
   Sample configuration:

```
pe11# show protocols mpls
label-switched-path toPE12-bronze {
    transport-class bronze;
}
label-switched-path toPE12-gold {
    transport-class gold;
}
```

Intra-AS BGP classful transport plane functionality:

- BGP classful transport creates predefined transport RIBs per named transport class (gold and bronze) and auto derives mapping community from its color value (100 and 200).
- Intra-AS transport routes are populated in transport RIBs by the tunneling protocol when it is associated with a transport class.
  In this example, RSVP LSP routes associated with transport class gold (color 100) and transport class bronze (color 200) are installed in the transport RIBs **junos-rti-tc-<100>.inet.3** and **junos-rti-tc-<200>.inet.3**, respectively.
- Service node (ingress PEs) match extended color community (color:0:100 and color:0:200) of service route against the mapping community in predefined resolution RIBs and resolve the protocol next hop (PNH) in corresponding transport RIBs (either junos-rti-tc-<100>.inet.3, or junos-rti-tc-<200>.inet.3).
- BGP routes bind to a resolution scheme by carrying the assiocaited mapping community.
- Each transport class automatically creates two predefined resolution schemes and automatically derives the mapping community.
  One resolution scheme is for resolving service routes that use **Color:0:<val>** as the mapping community.

  The other resolution scheme is for resolving transport routes that use **Transport-Target:0:<val>** as the mapping community.
- If service route PNH cannot be resolved using RIBs listed in the predefined resolution scheme, then it can fall back to the inet.3 routing table.

- You can also configure fallback between different colored transport RIBs by using user-defined resolution schemes under the **[edit routing-options resolution scheme]** configuration hierarchy.

Inter-AS Implementation of BGP Classful Transport Planes

In an inter-AS network, BGP-LU is converted to BGP classful transport network after configuring a minimum of two transport classes (gold and bronze) on all service nodes or PE devices and border nodes (ABRs and ASBRs).

To convert the transport tunnels into BGP classful transport:

1. Define transport class at the service nodes (ingress PE devices) and the border nodes (ABRs and ASBRs), for example, gold and broze.

   Sample configuration:

```
pe11# show routing-options
route-distinguisher-id 172.16.1.1;
transport-class {
    name gold {
        color 100;
    }
    name bronze {
        color 200;
```

2. Associate the transport tunnels to a specific transport class at the ingress node of the tunnels (ingress PEs, ABRs,and ASBRs).

   Sample configuration:

   For RSVP LSPs

```
abr23# show protocols mpls
label-switched-path toASBR21-bronze {
    transport-class bronze;
}
label-switched-path toASBR22-gold {
    transport-class gold;
```

   For IS-IS flxible algorithm

```
asbr13# show routing-options
flex-algorithm 128 {
```

```
…
color 100;
use-transport-class;
}
flex-algorithm 129 {
…
color 200;
use-transport-class;
}
```

3. Enable new family for the BGP classful transport (inet transport) and BGP-LU (inet labeled-unicast) in the network.

   Sample configuration:

```
abr23# show protocols bgp
group toAs2-RR27 {
    family inet {
        labeled-unicast {
…
        }
        transport {
…
    }
    cluster 172.16.2.3;
    neighbor 172.16.2.7;
}
```

4. Advertise service routes from the egress PE device with appropriate extended color community.

   Sample configuration:

```
pe11# show policy-options policy-statement red
term 1 {
        from {
            route-filter 192.168.3.3/32 exact;
        }
        then {
            community add map2gold;
            next-hop self;
            accept;
        }
```

```
    }
    term 2 {
        from {
            route-filter 192.168.33.33/32 exact;
        }
        then {
            community add map2bronze;
            next-hop self;
            accept;
        }
    }
community map2bronze members color:0:200;
community map2gold members color:0:100;
```

Inter-AS BGP classful transport plane functionality:

1. BGP classful transport planes create predefined transport RIBs per named transport class (gold and bronze) and automatically derives mapping community from its color value.
2. Intra-AS transport routes are populated in transport RIBs by tunneling protocols when associated with a transport class.

   For example, transport tunnel routes associated with the transport class gold and bronze are installed in the transport RIBs **junos-rti-tc-<100>.inet.3** and **junos-rti-tc-<200>.inet.3**, respectively.
3. BGP classful transport planes use unique Route Distinguisher and Route Target when it copies the transport tunnel routes from each transport RIB to the bgp.transport.3 routing table.
4. Border nodes advertise routes from bgp.transport.3 routing table to its peers in other domains if family inet transport is negotiated in the BGP session.
5. Receiving border node installs these bgp-ct routes in the bgp.transport.3 routing table and copies these routes based on the transport Route Target to the appropriate transport RIBs.
6. Service node matches the color community in the service route against a mapping community in the resolution schemes and resolves PNH in the corresponding transport RIB (either **junos-rti-tc-<100>.inet.3**, or **junos-rti-tc-<200>.inet.3**).
7. Border nodes use predefined resolution schemes for transport route PNH resolution.
8. Predefined or user defined, both resolution schemes support service route PNH resolution. Predefined uses inet.3 as fallback, and user-defined resolution scheme allows list of transport RIBs to be used in the order specified while resolving PNH.
9. If service route PNH cannot be resolved using RIBs listed in the user-defined resolution scheme, then route is discarded.

Example: Configuring Classful Transport Planes (Intra-Domain)

IN THIS SECTION

Before You Begin

Hardware and Software requirements	Junos OS Release 21.1R1 or later. **NOTE**: Only the provider edge routers (PE1 and PE2) require Junos OS Release support for the BGP-CT feature.
Estimated reading time	45 minutes
Estimated configuration time	1 hour

What to expect?	A working BGP-CT network with three service levels that map to diversely routed LSP paths. A Junos configuration that maps specific traffic (VPN customer routes) to the desired transport class using BGP color attribute extended communities. Basic LSP traffic engineering to force traffic classes on to diverse paths in the provider network

Business impact	Use this configuration example to configure and verify the BGP Classful Transport (BGP-CT) feature within a single autonomous network (intra-domain). BGP-CT maps customer routes to network paths that can be engineered to provide varying levels of performance. A typical use case for intra-domain BGP-CT is for a service provider to deploy BGP-CT to offer tiered VPN service levels to their customers.
Useful resources:	
Know more	To better understand BGP-CT, see BGP Classful Transport Planes Overview
Juniper vLabs	Visit the Juniper virtual labs (vLabs) to reserve a pre-configured sandbox. Use the sandbox to interact with and understand the BGP-CT feature. You'll find the "Classful Transport Planes (Intra-Domain Scenario)" demonstration in the routing section .
Learn more	Junos Class of Service (JCOS) On-Demand

Functional Overview

Table 2 on page 290 provides a quick summary of the configuration components deployed in this example.

Table 2: Classful Transport Planes Functional Overview

Routing and Signaling protocols	
OSPF	All routers run OSPF as the IGP. All routers belong to area 0 (also called the backbone area). The single OSPF routing domain provides loopback connectivity in the provider network.

Internal and External BGP	The customer edge (CE) devices use EBGP peering to exchange routes with their provider edge device as part of a Layer 3 VPN service. The PE devices use IBGP to exchange IPv4 Layer 3 VPN routes with the remote PE. These routes also carry a color community used to map traffic to the correct data plane tunnel. Our example does not use a route reflector, instead opting for direct PE-PE peering. **NOTE**: The provider router (P router) does not run BGP. Its part of a BGP-free core to promote scaling. The P device uses MPLS label based switching to transport the customer VPN traffic between the PE devices.
RSVP	Each PE devices signals three LSPs to the remote PE. These LSPs map to the corresponding service classes of gold, bronze, and Best-Effort (BE). RSVP supports rich traffic engineering to force traffic onto desired paths in the provider network. These paths can in turn be engineered to provide varying Class of Service (CoS) handling need to enforce the SLA for each transport class. Our basic topology provides three paths between the PE devices. We use a named path with an ERO to ensure diverse routing of the LSPs over the core. Junos supports a rich set of capabilities for traffic engineering. **NOTE**: The classful transport feature is also supported with LSPs established through segment routing–traffic engineering (SR-TE) and IS-IS flex-algorithm tunnels.
MPLS	The provider network uses a MPLS based label switching data plane. The use of MPLS with TE paths ensures that each service class can be routed over disjoint paths with the desired performance levels. As noted above, MPLS also provides support for a BGP-free core.
Transport tunnels	

Three MPLS tunnels (LSPs) are established between the PE devices:	Each tunnel is assigned to the following transport classes: • Gold • Bronze • Best-effort This is the default transport class. This class provides best-effort (BE) level service. Customers that are not mapped to any specific transport class, or those that are mapped to a transport class that is down, default to the BE service class and the associated LSP path.
Service family	
Layer 3 VPN (`family inet-vpn unicast`	BGP-CT also works with other service families, such as BGP labeled Unicast, Flowspec, or Layer 2 VPN.
Primary verification tasks	
• Confirm overall network operation.	Verify working of IGP, RSVP, MPLS, BGP, and L3VPN.
• Verify mapping of Layer 3 VPN customer traffic to a transport class.	Modify the network to effect traffic steering between transport class tunnels to simulate the failure of a service tunnel and subsequent fail over to the BE path/class.

Topology Overview

This configuration example is based on a simple MPLS-Based Layer 3 VPN with two customer edge (CE) devices that communicate over the service provider network. The network core has three provider (P) routers that transport the VPN customer traffic using labeled-based switching. The two provider edge (PE) devices provide a Layer 3 VPN service to their attached CEs. The PEs use RSVP signaled MPLS LSPs to transport VPN traffic over the core. See Example: Configure a Basic MPLS-Based Layer 3 VPN for background information on the operation and configuration of a MPLS-based L3VPN.

We focus on the left to right flow of traffic from CE1 to CE2, and how PE1 uses a BGP color community attached to routes learned from PE2 to map traffic sent to the remote CE over the desired LSP forwarding next hops. In our example, PE1 uses explicit route objects (ERO) to force the routing of these LSPs over diverse paths. We skip this step at PE2, allowing the LSPs to be routed based on IGP load balancing. In order to have traffic flow from CE1 to CE2, CE1 must have a route to reach CE2. The

routes for CE2 travel in the opposite direction of the traffic it attracts from CE1. That is, the route to CE2's loopback travels from right to left.

In our example, the gold service class LSP is constrained to the PE1-P1-PE2 path. The bronze service class uses the PE1-P2-PE2 path. The best-effort LSP is routed along the PE1-P3-PE2 path. The topology diagram uses colored links to represent the three paths.

In our example, we add the `protocols mpls icmp-tunneling` statement. This is to allow the CE devices to trace the path through the provider network, even when that path involves MPLS switching as is the case for the Layer 3 VPN traffic. This option helps you confirm the expected forwarding path as a function of transport class is used.

Table 3 on page 293 describes the role and functionality of each device in the context of this topology. Click on any device name to view its quick configuration.

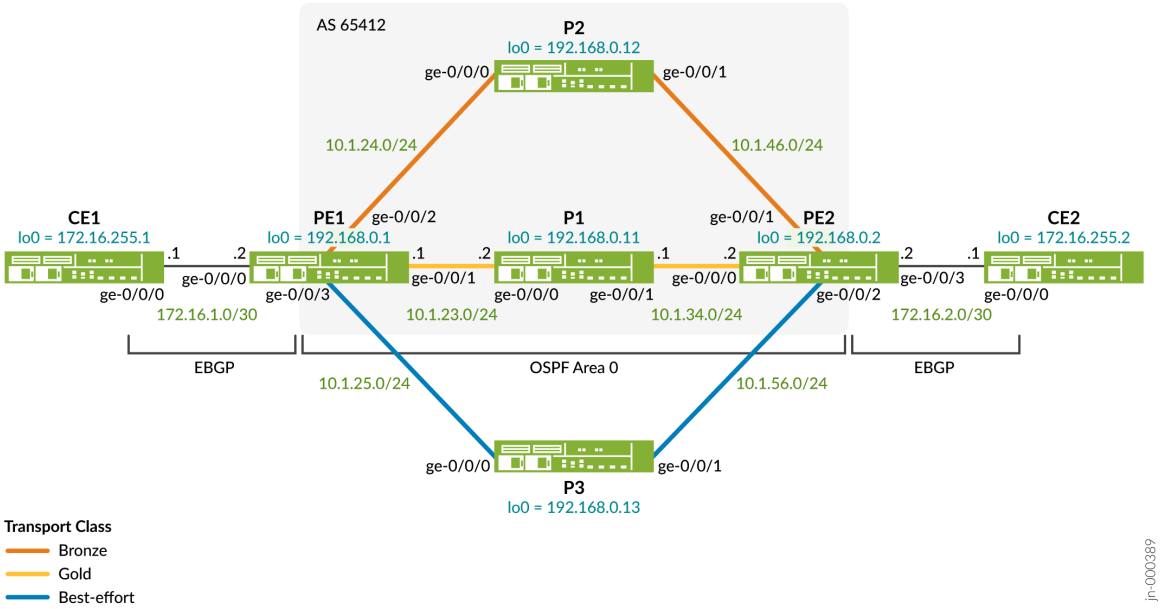
Table 3: Intra-Domain Classful Transport Planes Topology Overview

Device Name	Role	Function
"CE1" on page 317	Local CE device (R1) .	EBGP peer to PE1 router to advertise and learn CE device loopback addresses. Test service connectivity with pings to the loopback address of CE2.
"CE2" on page 318	Remote CE device (R7)	EBGP peering to PE2 router to advertise and learn CE device loopback addresses. Configures and attaches the color mapping community.
"PE1 (DUT)" on page 318	local PE device (R2).	PE1 maps the color tagged service routes that originate at CE2 to a cosponsoring transport class (TC). PE1 receives the color tags routes over its IBGP session to PE2. In this example PE1 uses ERO based constraints to force diverse routing of its three LSPs over the provider's core.

"PE2" on page 320	Remote PE device (R6).	PE2 re-advertises the color tagged routes received by CE2 to PE1 using IBGP. These routes uses the `inet-vpn` family to support a Layer 3 VPN services with color mapped TCs.
"P1" on page 321"P2" on page 322"P3" on page 322	Provider devices P1, P2, and P3 (R3, R4, and R5).	The P1-P3 devices represent the service provider's core network. These are pure transit devices that perform MPLS label switching to transport the CE traffic sent over the L3 VPN.

Topology Illustrations

Figure 25: Service Mapping Using Classful Transport Planes Within a Network Domain



PE1 Configuration Steps

For information about navigating the CLI, see Using the CLI Editor in Configuration Mode

NOTE: For complete configuration on all devices see:

- "Appendix 2: Set Commands on All Devices" on page 317
- "Appendix 3: Show Configuration Output on PE1" on page 323

This section highlights the main configuration tasks needed to configure the PE1 device for this example. The first step is common to configuring a basic Layer 3 VPN service. The following set of steps are specific to adding the BGP-CT feature to your Layer 3 VPN. Both PE devices have a similar configuration, here we focus on PE1.

1. First, provision the general Layer 3 VPN:

   a. Configure and number the loopback, core facing, and CE-facing interfaces for IPv4. Be sure to enable the `mpls` family on the core-facing interfaces connecting to the P devices to support MPLS switching.

   b. Configure an autonomous system number.

   c. Configure single area OSPF on the loopback and core-facing interfaces.

   d. Configure RSVP on the loopback and core-facing interfaces.

   e. Configure the IBGP peering session to the remote PE device, PE2. Include the `inet-vpn` address family to support an IPv4 Layer 3 VPN.

   f. Configure a VRF based routing-instance for the CE1 device. Use EBGP as the PE-CE routing protocol.

```
[edit]
set interfaces ge-0/0/1 unit 0 family inet address 10.1.23.1/24
set interfaces ge-0/0/1 unit 0 family mpls

set interfaces ge-0/0/2 unit 0 description "Link from PE1 to P2"
set interfaces ge-0/0/2 unit 0 family inet address 10.1.24.1/24
set interfaces ge-0/0/2 unit 0 family mpls

set interfaces ge-0/0/3 unit 0 description "Link from PE1 to P3"
set interfaces ge-0/0/3 unit 0 family inet address 10.1.25.1/24
set interfaces ge-0/0/3 unit 0 family mpls
```

```
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
```

```
[edit]
set routing-instances CE1_L3vpn instance-type vrf
set routing-instances CE1_L3vpn protocols bgp group CE1 type external
set routing-instances CE1_L3vpn protocols bgp group CE1 peer-as 64510
set routing-instances CE1_L3vpn protocols bgp group CE1 neighbor 172.16.1.1
set routing-instances CE1_L3vpn interface ge-0/0/0.0
set routing-instances CE1_L3vpn route-distinguisher 192.168.0.1:12
set routing-instances CE1_L3vpn vrf-target target:65412:12
```

```
[edit]
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.1
set protocols bgp group ibgp family inet unicast
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 192.168.0.2

set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0

set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface ge-0/0/2.0
set protocols rsvp interface ge-0/0/3.0
```

```
[edit]
set routing-options route-distinguisher-id 192.168.0.1
set routing-options autonomous-system 65412
```

2. Add classful transport planes to the Layer 3 VPN.

   Configure the gold and bronze transport-classes.

This is a critical step in configuring the classful transport feature. These transport classes are mapped to RSVP signaled (and possibly traffic engineered) LSPs that traverse the provider core. The remote routes learned from CE2 are tagged with color communities that map to these transport classes, and in so doing, to the desired LSP between the PE devices.

```
[edit]
set routing-options transport-class name gold color 100
set routing-options transport-class name bronze color 200
set routing-options resolution preserve-nexthop-hierarchy
```

3. Configure three LSPs from PE1 to PE2 with constrained routing that forces each to traverse a different P router. Two of these LSPs map to the *gold* and *bronze* transport-classes. The gold LSP is routed through P1, the bronze through P2, and the best-effort through the P3 device.

   Once mapped to transport classes the service provider is able to place specific customer traffic, as indicated by a BGP color community, onto a specific LSP. With this color to LSP mapping the service provider can offer a tiered service with different SLAs.

   In our example we use a strict ERO to ensure the three LSPs are diversely routed over the three paths available in the topology.

```
[edit]
set protocols mpls label-switched-path lsp_to_pe2 to 192.168.0.2
set protocols mpls label-switched-path lsp_to_pe2 primary best-effort
set protocols mpls label-switched-path gold_lsp_to_pe2 to 192.168.0.2
set protocols mpls label-switched-path gold_lsp_to_pe2 preference 5
set protocols mpls label-switched-path gold_lsp_to_pe2 primary gold
set protocols mpls label-switched-path gold_lsp_to_pe2 transport-class gold
set protocols mpls label-switched-path bronze_lsp_to_pe2 to 192.168.0.2
set protocols mpls label-switched-path bronze_lsp_to_pe2 preference 5
set protocols mpls label-switched-path bronze_lsp_to_pe2 primary bronze
set protocols mpls label-switched-path bronze_lsp_to_pe2 transport-class bronze
set protocols mpls path gold 10.1.23.2 strict
set protocols mpls path bronze 10.1.24.2 strict
set protocols mpls path best-effort 10.1.25.2 strict
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set protocols mpls interface ge-0/0/3.0
```

4. To facilitate fallback to the default service class (best-effort) tunnel, we configure the gold and bronze transport class tunnels with a lower global preference. In this example the preference value is changed from the default 7 to 5. This allows the use of the best-effort tunnel as a fallback in the

event the gold or bronze tunnels become unusable. Setting a lower (more preferred) preference on the gold and bronze tunnels ensures they are selected for forwarding, even though the service route is able to resolve to the best-effort tunnel.

> **NOTE**: This section focused on the configuration needed on the PE1 device. It should be noted that for theclassful transport next-hop selection function to work at PE1, the remote CE2 device routes must be tagged with a color community. This tagging can occur on the remote PE2 device, or on the remote CE2 device. We show the latter approach here for completeness:

5. Match the color community tags added at the remote CE2 to the transport class definitions for the bronze and gold TCs.

```
[edit]
set policy-options policy-statement adv_direct term 1 from protocol direct
set policy-options policy-statement adv_direct term 1 from route-filter 172.16.0.0/16 orlonger
set policy-options policy-statement adv_direct term 1 then community add map2bronze
set policy-options policy-statement adv_direct term 1 then accept
set policy-options community map2bronze members color:0:200
set policy-options community map2gold members color:0:100
```

Verify Classful Transport Planes

IN THIS SECTION

> **NOTE**: In this section we focus on commands that demonstrate a working classful transport feature. See "Appendix 1: Troubleshooting" on page 308 for commands used to verify the underlying functionality needed by the classful transport feature.

You'll use these commands to verify BGP classful transport works correctly.

Table 4: Classful Transport Planes Verification Commands

Command	Verification Task
show route resolution scheme	Display how service class routes are resolved to LSP next hops. Verify the resolution routing tables for a specific route.
show route receiving-protocol bgp *pe2-loopback-address*	Verify that the VPN routes received by PE1 have a BGP color community attached.
show route and show route forwarding-table vpn *vpn*	Verify transport tunnel selection by viewing the protocol nexthop (PNH) for the routes at PE1.
show mpls lsp statistics and show route forwarding-table	Verify the transport tunnel used by a specific transport class route.

Verify Transport Classes and Transport Tunnels

Purpose

PE1 and PE2 use RSVP-signaled MPLS transport tunnels to support a Layer 3 VPN service that is capable of offering differentiated service levels. These service routes have their next hops resolved to a specific MPLS tunnel based on the corresponding service class. The service class is signaled by attaching a BGP color community to VPN customer routes.

In this part you confirm that all three of PE1's LSPs are operational, that they are mapped to the correct transport class, and that they are correctly routed over the provider's core.

Action

From operational mode, enter the `show route 192.168.0.2` command.

```
user@PE1 show route 192.168.0.2
inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.2/32     *[OSPF/10] 00:27:20, metric 2
                       to 10.1.24.2 via ge-0/0/2.0
```

```
                    >  to 10.1.25.2 via ge-0/0/3.0
                       to 10.1.23.2 via ge-0/0/1.0

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

192.168.0.2/32     *[RSVP/7/1] 00:13:09, metric 2
                    >  to 10.1.25.2 via ge-0/0/3.0, label-switched-path lsp_to_pe2

junos-rti-tc-100.inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.2/32     *[RSVP/5/1] 00:13:11, metric 2
                    >  to 10.1.23.2 via ge-0/0/1.0, label-switched-path gold_lsp_to_pe2

junos-rti-tc-200.inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.2/32     *[RSVP/5/1] 00:13:08, metric 2
                    >  to 10.1.24.2 via ge-0/0/2.0, label-switched-path bronze_lsp_to_pe2
```

Meaning

The output confirms that PE1 has learned three paths to the loopback of PE2 through OSPF. These routes are in the `inet.0` table. You also note that all three LSPs are represented as viable next hops to reach PE2. Note that each of these LSPs is housed in a different routing table. The highlighted portion of the IP next hops (and the corresponding interface name) confirms the desired diverse LSP routing over the core. Traffic mapped to the gold path is sent to 10.1.23.2, while traffic for bronze and BE is sent to 10.1.24.2 and 10.1.25.2, respectively.

The following transport RIBs and transport tunnels are created.

- `junos-rti-tc-100.inet.3` for gold_lsp_to_pe2
- `junos-rti-tc-200.inet.3` for bronze_lsp_to_pe2
- `inet.3` for lsp_to_pe2

Verify Next Hop Resolution Schemes

Purpose

Verify the service routes resolution schemes, the associated mapping community, and how the next hop resolves over the contributing routing tables.

Action

From operational mode, enter the `show route resolution scheme all` command.

```
user@PE1> show route resolution scheme all
Resolution scheme: junos-resol-schem-tc-100-v4-service
  References: 1
  Mapping community: color:0:100
  Resolution Tree index 1, Nodes: 1
  Policy: [__resol-schem-common-import-policy__]
  Contributing routing tables: junos-rti-tc-100.inet.3 inet.3

Resolution scheme: junos-resol-schem-tc-100-v4-transport
  References: 1
  Mapping community: transport-target:0:100
  Resolution Tree index 3, Nodes: 1
  Policy: [__resol-schem-common-import-policy__]
  Contributing routing tables: junos-rti-tc-100.inet.3

Resolution scheme: junos-resol-schem-tc-100-v6-service
  References: 1
  Mapping community: color:0:100
  Resolution Tree index 2, Nodes: 0
  Policy: [__resol-schem-common-import-policy__]
  Contributing routing tables: junos-rti-tc-100.inet6.3 inet6.3

Resolution scheme: junos-resol-schem-tc-100-v6-transport
  References: 1
  Mapping community: transport-target:0:100
  Resolution Tree index 4, Nodes: 0
  Policy: [__resol-schem-common-import-policy__]
  Contributing routing tables: junos-rti-tc-100.inet6.3

Resolution scheme: junos-resol-schem-tc-200-v4-service
  References: 1
```

```
  Mapping community: color:0:200
  Resolution Tree index 5, Nodes: 1
  Policy: [__resol-schem-common-import-policy__]
  Contributing routing tables: junos-rti-tc-200.inet.3 inet.3

Resolution scheme: junos-resol-schem-tc-200-v4-transport
  References: 1
  Mapping community: transport-target:0:200
  Resolution Tree index 7, Nodes: 1
  Policy: [__resol-schem-common-import-policy__]
  Contributing routing tables: junos-rti-tc-200.inet.3

Resolution scheme: junos-resol-schem-tc-200-v6-service
  References: 1
  Mapping community: color:0:200
  Resolution Tree index 6, Nodes: 0
  Policy: [__resol-schem-common-import-policy__]
  Contributing routing tables: junos-rti-tc-200.inet6.3 inet6.3

Resolution scheme: junos-resol-schem-tc-200-v6-transport
  References: 1
  Mapping community: transport-target:0:200
  Resolution Tree index 8, Nodes: 0
  Policy: [__resol-schem-common-import-policy__]
  Contributing routing tables: junos-rti-tc-200.inet6.3
```

Meaning

Focusing on the IPv4 portions of the output, you see the **junos-tc-100 (gold)** transport class has two resolution schemes - **junos-resol-schem-tc-100-v4-service** and **junos-resol-schem-tc-100-v4-transport** – used for the service and transport routes, respectively.

The resolution scheme for gold service routes (**junos-resol-schem-tc-100-v4-service**) provides resolution over *both* the **junos-rti-tc-100.inet.3** and the `inet.3` routing tables (highlighted in the sample output). Listing both the service and BE resolution tables is how fallback occurs when the service class is down. Recall this is why we altered the preference value of the service LSPs (setting the preference to 5 rather than the default 7), to ensure the service route is always preferred over the BE fallback.

Verify Color Tagging and Next Hop Selection for CE2 Routes

Purpose

Confirm that PE2 advertises the loopback route for CE2 with a color community that selects the bronze service class (color 200).

NOTE: In our example we configure the CE2 device to attach the color community. PE2 leaves this community in place when it re-advertises the route to PE1. This means the VPN customer is able to effect their own service class mappings. When desired the PE router can bleach or strip out any communities received from the CE. In this case the PE device needs to be configured to attach the desired color mapping community to CE routes before it re advertises them to PE1.

Action

From operational mode, enter the `show route receive-protocol bgp 192.168.0.2 172.16.255.2 detail` command.

```
user@PE1> show route receive-protocol bgp 192.168.0.2 172.16.255.2 detail
inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)

CE1_L3vpn.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
* 172.16.255.2/32 (1 entry, 1 announced)
     Import Accepted
     Route Distinguisher: 192.168.0.2:12
     VPN Label: 299808
     Nexthop: 192.168.0.2
     Localpref: 100
     AS path: 64520 I
     Communities: target:65412:12 color:0:200
```

Display the forwarding table entry for the CE2 loopback in the VPN routing instance at PE1. Confirm the forwarding next hop matches the desired transport class (bronze). Use the `show route forwarding-table vpn CE1_L3vpn destination 172.16.255.2 extensive` command.

```
user@PE1> show route forwarding-table vpn CE1_L3vpn destination 172.16.255.2 extensive
Routing table: CE1_L3vpn.inet [Index 10]
Internet:

Destination:  172.16.255.2/32
```

```
  Route type: user
  Route reference: 0                   Route interface-index: 0
  Multicast RPF nh index: 0
  P2mpidx: 0
  Flags: sent to PFE, prefix load balance
  Next-hop type: indirect              Index: 1048574  Reference: 2
  Nexthop:
  Next-hop type: composite             Index: 662      Reference: 2
  Load Balance Label: Push 299808, None
  Nexthop: 10.1.24.2
  Next-hop type: Push 299872           Index: 653      Reference: 2
  Load Balance Label: None
  Next-hop interface: ge-0/0/2.0
```

Meaning

The highlighted entries confirm traffic matching the CE2 loopback route is sent to 10.1.24.2 using the ge-0/0/2 interface. Recall from the EROs used for the LSPs, this interface and next hop is associated with the bronze LSP and transport class. The `299808` label is used to identify the service VRF. The outer RSVP transport label is `299872`.

You quickly confirm this is the correct RSVP transport label for the bronze class with a `show rsvp session detail name bronze_lsp_to_pe2` command

```
root@PE1> show rsvp session detail name bronze_lsp_to_pe2
Ingress RSVP: 3 sessions

192.168.0.2
  From: 192.168.0.1, LSPstate: Up, ActiveRoute: 0
  LSPname: bronze_lsp_to_pe2, LSPpath: Primary
  LSPtype: Static Configured
  Suggested label received: -, Suggested label sent: -
  Recovery label received: -, Recovery label sent: 299872
  Resv style: 1 FF, Label in: -, Label out: 299872
  Time left:    -, Since: Tue Aug 16 12:17:12 2022
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 23256 protocol 0
  PATH rcvfrom: localclient
  Adspec: sent MTU 1500
  Path MTU: received 1500
  PATH sentto: 10.1.24.2 (ge-0/0/2.0) 1 pkts
  RESV rcvfrom: 10.1.24.2 (ge-0/0/2.0) 1 pkts, Entropy label: Yes
```

```
  Explct route: 10.1.24.2 10.1.46.2
  Record route: <self> 10.1.24.2 10.1.46.2
Total 1 displayed, Up 1, Down 0
```

The highlighted portions call out that the bronze LSP is routed through the P2 device and is associated with the indicated RSVP transport label (`299856`) you previously confirmed in the VPN forwarding table for the CE2 loopback address.

Verify End-to-End Connectivity

Purpose

Verify end-to-end connectivity across the provider's domain by pinging between CE1 to CE2. You examine MPLS traffic statistics to provide additional confirmation that the bronze transport class is used.

Action

From operational mode, enter the `ping` command.

```
user@CE1> ping 172.16.255.2 source 172.16.255.1 count 100 rapid
PING 172.16.255.2 (172.16.255.2): 56 data bytes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!
--- 172.16.255.2 ping statistics ---
100 packets transmitted, 100 packets received, 0% packet loss
round-trip min/avg/max/stddev = 2.647/3.589/30.264/2.695 ms
```

From operational mode at PE1, enter the `show mpls lsp statistics` command.

```
user@PE1> show mpls lsp statistics
Ingress LSP: 3 sessions
To              From            State     Packets            Bytes LSPname
192.168.0.2     192.168.0.1     Up            100



                  8400 bronze_lsp_to_pe2
192.168.0.2     192.168.0.1     Up              0                0 gold_lsp_to_pe2
192.168.0.2     192.168.0.1     Up              0                0 lsp_to_pe2
```

```
Total 3 displayed, Up 3, Down 0
<output truncated for brevity>
```

Action

Trace the route from CE1 to CE2's loopback. Our configuration includes the `icmp-tunneling` statement to support an ICMP based trace route with MPLS hops in the provider core.

```
user@CE1> traceroute no-resolve 172.16.255.2
traceroute to 172.16.255.2 (172.16.255.2), 30 hops max, 52 byte packets
 1  172.16.1.2  2.174 ms  1.775 ms  1.917 ms
 2  10.1.24.2  5.171 ms  5.768 ms  4.900 ms
     MPLS Label=299872 CoS=0 TTL=1 S=0
     MPLS Label=299808 CoS=0 TTL=1 S=1
 3  10.1.46.2  4.707 ms  4.347 ms  4.419 ms
     MPLS Label=299808 CoS=0 TTL=1 S=1
 4  172.16.255.2  5.640 ms  5.851 ms  44.777 ms
```

Meaning

The ping exchange is successful and the statistics confirm use of the bronze transport tunnel. This is expected given the route to CE2 has the 200 color community attached. The trace route results confirm the traffic is forwarded over a LSP, and that this LSP is forwarding through 10.1.24.2. This is the IP address assigned to the P2 device. The forwarding next hop and outer label value confirm this traffic is sent on the bronze service class LSP.

Confirm Fail Over to Best-Effort

Purpose

Bring the bronze transport LSP down to verify that the traffic sent to CE2 fails over to the BE path.

Action

Enter configuration mode and specify an invalid next hop as an ERO for the bronze transport tunnel. The inability to satisfy the ERO requirement brings the related LSP down.

```
[edit]
user@PE1# set protocols mpls path bronze 10.1.66.6 strict
```

Once the change is committed the bronze tunnel is shown down:

```
root@PE1> show mpls lsp ingress
Ingress LSP: 3 sessions
To              From            State Rt P     ActivePath       LSPname
192.168.0.2     0.0.0.0         Dn     0       -                bronze_lsp_to_pe2
192.168.0.2     192.168.0.1     Up     0 *     gold             gold_lsp_to_pe2
192.168.0.2     192.168.0.1     Up     0 *     best-effort      lsp_to_pe2
```

Repeat the `ping` and trace route commands from CE1 to CE2's loopback.

```
root@CE1> ping 172.16.255.2 source 172.16.255.1 count 100 rapid
PING 172.16.255.2 (172.16.255.2): 56 data bytes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!
--- 172.16.255.2 ping statistics ---
100 packets transmitted, 100 packets received, 0% packet loss
round-trip min/avg/max/stddev = 4.164/5.345/12.348/1.240 ms

root@CE1> traceroute no-resolve 172.16.255.2
traceroute to 172.16.255.2 (172.16.255.2), 30 hops max, 52 byte packets
 1  172.16.1.2  2.493 ms  1.766 ms  1.913 ms
 2  10.1.25.2  5.211 ms  5.016 ms  5.514 ms
     MPLS Label=299808 CoS=0 TTL=1 S=0
     MPLS Label=299808 CoS=0 TTL=1 S=1
 3  10.1.56.2  4.216 ms  4.467 ms  4.551 ms
     MPLS Label=299808 CoS=0 TTL=1 S=1
 4  172.16.255.2  5.492 ms  5.543 ms  5.112 ms
```

Again display MPLS statistics on PE1.

```
user@PE1> show mpls lsp statistics

root@PE1> show mpls lsp statistics
Ingress LSP: 3 sessions
To              From            State     Packets            Bytes LSPname
192.168.0.2     0.0.0.0         Dn             NA               NA bronze_lsp_to_pe2
192.168.0.2     192.168.0.1     Up              0                0 gold_lsp_to_pe2
192.168.0.2     192.168.0.1     Up            100             8400 lsp_to_pe2
```

```
Total 3 displayed, Up 2, Down 1
. . .
```

Meaning

The ping exchange still succeeds, albeit now on a best-effort path. On the PE the statistics confirm use of the best-effort transport tunnel. The trace route shows that PE1 now forwards to the 10.1.25.2 next hop through PE3. This confirms fail over from a colored transport class to the best-effort class in the event of transport tunnel failure.

NOTE: In this section we effected fail over to the BE class by bringing down the LSP mapped to the bronze service class. As an alternative, consider changing the EBGP export policy on the CE2 device to have it attach the gold (100) color community. With this approach you expect to see ping traffic from CE1 to CE2 taking the gold LSP rather than failing over to BE. The below does the trick at CE2 if you prefer this approach. Be sure to commit the changes at CE2.

```
[edit]
root@CE2# delete policy-options policy-statement adv_direct term 1 then community add
map2bronze
root@CE2# set policy-options policy-statement adv_direct term 1 then community add
map2gold
```

Appendix 1: Troubleshooting

Our verification section is based on an assumption that you have a working network, allowing the focus to be placed on confirming the operation of BGP-CT. The BGP-CT feature, in a MPLS-based Layer 3 VPN context, is dependent on a network with working interfaces, IGP, RSVP, MPLS, and BGP.

Table 5 on page 309 provides guidance on what to look for if your BGP-CT solution is not working as expected. The table is structured from the bottom to the top, starting with basic interface connectivity and ending with successful BGP route exchange between the PE devices.

NOTE: As part of this example you configure a loopback address and router ID. If the device previously had a different RID it can take some time for things to stabilize. Changing the RID is very disruptive and not something that happens often. When in a lab environment its suggested that you issue the `restart routing` operational mode command on all devices right after committing the new RID.

Table 5: Troubleshooting Steps

Functional Layer	Verification Approach
Interfaces and IP addressing	Verify that all interfaces in your topology are operationally up. Verify you can ping both the local and remote end of each link. Like most networks, the protocols and services in this example require a working IPv4 infrastructure. root@PE1> **show interfaces terse | match "(ge-0/0/0|ge-0/0/1| ge-0/0/2|ge-0/0/3)"** ge-0/0/0 up up ge-0/0/0.0 up up inet 172.16.1.2/30 ge-0/0/1 up up ge-0/0/1.0 up up inet 10.1.23.1/24 ge-0/0/2 up up ge-0/0/2.0 up up inet 10.1.24.1/24 ge-0/0/3 up up ge-0/0/3.0 up up inet 10.1.25.1/24 root@PE1> **ping 10.1.23.2 count 1** PING 10.1.23.2 (10.1.23.2): 56 data bytes 64 bytes from 10.1.23.2: icmp_seq=0 ttl=64 time=2.951 ms --- 10.1.23.2 ping statistics --- 1 packets transmitted, 1 packets received, 0% packet loss round-trip min/avg/max/stddev = 2.951/2.951/2.951/0.000 ms root@PE1> **ping 172.16.1.1 routing-instance CE1_L3vpn count 1** PING 172.16.1.1 (172.16.1.1): 56 data bytes 64 bytes from 172.16.1.1: icmp_seq=0 ttl=64 time=2.755 ms --- 172.16.1.1 ping statistics --- 1 packets transmitted, 1 packets received, 0% packet loss round-trip min/avg/max/stddev = 2.755/2.755/2.755/0.000 ms

OSPF (IGP) Routing	Confirm all provider devices have all expected OSPF adjacencies. Use the `show ospf interfaces` and `show ospf neighbors` operational mode commands. Display the routes for the provider loopback addresses and confirm valid OSPF paths for all remote loopback addresses (`show route protocol ospf | match 192.168.0`). Ping from the local loopback to the remote loopback addresses of all provider routers.
	This example uses CSPF based LSPs. This requires that OSPF be configured with the `traffic-engieering` statement. If IS-IS is used as the IGP this statement is not needed.
	
	`root@PE1> **show ospf interface**`
	`Interface State Area DR ID BDR ID Nbrs`
	`ge-0/0/1.0 BDR 0.0.0.0 192.168.0.11 192.168.0.1 1`
	`ge-0/0/2.0 BDR 0.0.0.0 192.168.0.12 192.168.0.1 1`
	`ge-0/0/3.0 DR 0.0.0.0 192.168.0.1 192.168.0.13 1`
	`lo0.0 DRother 0.0.0.0 0.0.0.0 0.0.0.0 0`
	
	`root@PE1> **show ospf neighbor**`
	`Address Interface State ID Pri Dead`
	`10.1.23.2 ge-0/0/1.0 **Full** 192.168.0.11 128 34`
	`10.1.24.2 ge-0/0/2.0 **Full** 192.168.0.12 128 32`
	`10.1.25.2 ge-0/0/3.0 **Full** 192.168.0.13 128 37`
	
	`root@PE1> **show route protocol ospf| match 192.168.0**`
	`192.168.0.2/32 *[OSPF/10] 00:10:15, metric 2`
	`192.168.0.11/32 *[OSPF/10] 00:18:40, metric 1`
	`192.168.0.12/32 *[OSPF/10] 00:18:35, metric 1`
	`192.168.0.13/32 *[OSPF/10] 00:10:15, metric 1`
	`root@PE1> **ping 192.168.0.2 source 192.168.0.1 count 1**`
	`PING 192.168.0.2 (192.168.0.2): 56 data bytes`
	`64 bytes from 192.168.0.2: icmp_seq=0 ttl=63 time=3.045 ms`
	
	`--- 192.168.0.2 ping statistics ---`

	1 packets transmitted, 1 packets received, 0% packet loss round-trip min/avg/max/stddev = 3.045/3.045/3.045/0.000 ms

MPLS and RSVP	Verify all core interfaces are enabled for the `mpls` family. with a `show interfaces terse` command. Also verify that all provider interfaces are enabled under the `protocols mpls` and `protocols RSVP` hierarchies. Use the `show mpls interfaces` and `show rsvp interfaces` commands.
	NOTE: Be sure to confirm that the correct interface unit numbers are listed for the MPLS family and for each protocol. This example uses unit 0, which is the default unit number, on all interfaces.
	
	`root@PE1> **show rsvp interface**`
	`RSVP interface: 4 active`
	`Active Subscr- Static`
	`Available Reserved Highwater`
	`Interface State resv iption BW`
	`BW BW mark`
	`ge-0/0/1.0 Up 1 100% 1000Mbps`
	`1000Mbps 0bps 0bps`
	`ge-0/0/2.0 Up 1 100% 1000Mbps`
	`1000Mbps 0bps 0bps`
	`ge-0/0/3.0 Up 1 100% 1000Mbps`
	`1000Mbps 0bps 0bps`
	`lo0.0 Up 0 100% 0bps`
	`0bps 0bps 0bps`
	
	`root@PE1> **show mpls interface**`
	`Interface State Administrative groups (x: extended)`
	`ge-0/0/1.0 Up <none>`
	`ge-0/0/2.0 Up <none>`
	`ge-0/0/3.0 Up <none>`
	
	On the PE routers confirm that the LSPs are correctly defined to egress at the remote PE device's loopback address. Verify the EROs and any other TE constraints are valid. Use the `show mpls lsp` and `show rsvp session` commands.
	NOTE: Our examples uses CSPF based LSPs. This requires that the IGP supports a TE database (TED). If OSPF is the IGP be sure to include the `traffic-engieering` configuration statement. Alternatively, consider using the `no-cspf` statement in the LSP definition to remove CSPF from the equation.
	
	`root@PE1> **show mpls lsp**`
	`Ingress LSP: 3 sessions`
	`To From State Rt P ActivePath`

```
LSPname
192.168.0.2     192.168.0.1     Up      0 *     bronze
bronze_lsp_to_pe2
192.168.0.2     192.168.0.1     Up      0 *     gold
gold_lsp_to_pe2
192.168.0.2     192.168.0.1     Up      0 *     best-effort
lsp_to_pe2
Total 3 displayed, Up 3, Down 0

Egress LSP: 3 sessions
To              From            State   Rt Style Labelin
Labelout LSPname
192.168.0.1     192.168.0.2     Up       0  1 FF       3
- bronze_lsp_to_pe1
192.168.0.1     192.168.0.2     Up       0  1 FF       3
- gold_lsp_to_pe1
192.168.0.1     192.168.0.2     Up       0  1 FF       3
- lsp_to_pe1
Total 3 displayed, Up 3, Down 0

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

BGP	Use the `show bgp summary`command on the PE devices to confirm both their EBGP session to the CE, and the IBGP session to the remote PE are established. If BGP is down despite being able to ping, suspect bad peer definition. Recall that loopback peering (for IBGP) requires the `local-address`statement. For EBGP specify directly connected next hops and confirm the local AS number, under `edit routing-options` and the remote AS number, under the EBGP peer group, are specified. Confirm the PE-PE session has the `inet-vpn unicast` family enabled. Use the `show route` command to confirm receipt of the remote CE route on the local PE. Use the `detail` switch to confirm proper color community attachment. root@PE1> **show bgp summary** Threading mode: BGP I/O Default eBGP mode: advertise - accept, receive - accept Groups: 2 Peers: 2 Down peers: 0 Table Tot Paths Act Paths Suppressed History Damp State Pending inet.0 0 0 0 0 0 0 bgp.l3vpn.0 2 2 0 0 0 0 Peer AS InPkt OutPkt OutQ Flaps Last Up/Dwn State|#Active/Received/Accepted/Damped... **172.16.1.1** 64510 55 55 0 0 23:13 **Establ** CE1_L3vpn.inet.0: 1/2/2/0 **192.168.0.2** 65412 57 56 0 0 23:11 **Establ** inet.0: 0/0/0/0 **bgp.l3vpn.0: 2/2/2/0** **CE1_L3vpn.inet.0: 2/2/2/0** The show route advertising and receiving protocol commands are useful when confirming what routes a given BGP speaker advertises or receives, respectively. root@PE1> **show route advertising-protocol bgp 192.168.0.2** CE1_L3vpn.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden) Prefix Nexthop MED

```
Lclpref    AS path
* 172.16.1.0/30           Self
100        I
* 172.16.255.1/32         Self
100        64510 I

root@PE1> show route receive-protocol bgp 192.168.0.2

inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0
hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

CE1_L3vpn.inet.0: 5 destinations, 6 routes (5 active, 0
holddown, 0 hidden)
  Prefix                  Nexthop              MED
Lclpref    AS path
* 172.16.2.0/30           192.168.0.2
100        I
* 172.16.255.2/32         192.168.0.2
100        64520 I

junos-rti-tc-100.inet.3: 1 destinations, 1 routes (1 active, 0
holddown, 0 hidden)

junos-rti-tc-200.inet.3: 1 destinations, 1 routes (1 active, 0
holddown, 0 hidden)

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
  Prefix                  Nexthop              MED
Lclpref    AS path
  192.168.0.2:12:172.16.2.0/30
*                         192.168.0.2
100        I
  192.168.0.2:12:172.16.255.2/32
*                         192.168.0.2
100        64520 I
```

Layer 3 VPN	Ensure that the IBGP session supports `family inet-vpn` routes. Confirm the routes advertised by remote PE are imported into the correct instance based on the route target. Ensure the import and export policy used in the routing instance of each PE match on and advertise the correct routes. Some of the displays in the BGP verification section confirm the receipt of remote CE routes and the importation of those routes into the VRF instance.
	
	root@PE1> **show bgp neighbor 192.168.0.2 | match nlri**
	  NLRI for restart configured on peer: inet-unicast inet-vpn-unicast
	  NLRI advertised by peer: inet-unicast inet-vpn-unicast
	  NLRI for this session: inet-unicast inet-vpn-unicast
	root@PE1> show route table CE1_L3vpn.inet
	
	root@PE1> **show route receive-protocol bgp 192.168.0.2 172.16.255.2 detail**
	. . .
	CE1_L3vpn.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
	* 172.16.255.2/32 (1 entry, 1 announced)
	  Import Accepted
	  Route Distinguisher: 192.168.0.2:12
	  VPN Label: 299776
	  Nexthop: **192.168.0.2**
	  Localpref: 100
	  AS path: 64520 I
	  Communities: **target:65412:12 color:0:200**
	
	root@PE1> **show route table CE1_L3vpn.inet**
	
	CE1_L3vpn.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
	+ = Active Route, - = Last Active, * = Both
	
	172.16.1.0/30 *[Direct/0] 00:30:11
	  > via ge-0/0/0.0
	  [BGP/170] 00:29:57, localpref 100
	  AS path: 64510 I, validation-state: unverified
	  > to 172.16.1.1 via ge-0/0/0.0
	172.16.1.2/32 *[Local/0] 00:30:11
	  Local via ge-0/0/0.0
	172.16.2.0/30 *[BGP/170] 00:21:26, localpref 100, from 192.168.0.2
	  AS path: I, validation-state: unverified

	> to 10.1.25.2 **via ge-0/0/3.0, label-switched-path lsp_to_pe2** 172.16.255.1/32 *[BGP/170] 00:29:57, localpref 100 AS path: 64510 I, validation-state: unverified > to 172.16.1.1 via ge-0/0/0.0 **172.16.255.2/32** *[BGP/170] 00:29:40, localpref 100, from 192.168.0.2 AS path: 64520 I, validation-state: unverified > to 10.1.24.2 via ge-0/0/2.0, **label-switched-path bronze_lsp_to_pe2** Confirm the CE device is receiving and advertising the expected routes using the methods discussed for BGP troubleshooting.

Appendix 2: Set Commands on All Devices

IN THIS SECTION

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

CE1

```
set interfaces ge-0/0/0 unit 0 description "Link from CE1 to PE1 for Layer 3 VPN"
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.1/30
set interfaces lo0 unit 0 family inet address 172.16.255.1/32
```

```
set policy-options policy-statement adv_direct term 1 from protocol direct
set policy-options policy-statement adv_direct term 1 from route-filter 172.16.0.0/16 orlonger
set policy-options policy-statement adv_direct term 1 then accept
set protocols bgp group ToPE1 type external
set protocols bgp group ToPE1 export adv_direct
set protocols bgp group ToPE1 peer-as 65412
set protocols bgp group ToPE1 neighbor 172.16.1.2
set routing-options router-id 172.16.255.1
set routing-options autonomous-system 64510
set system host-name CE1
```

CE2

```
set interfaces ge-0/0/0 unit 0 description "Link from CE2 to PE2 for Layer 3 VPN"
set interfaces ge-0/0/0 unit 0 family inet address 172.16.2.1/30
set interfaces lo0 unit 0 family inet address 172.16.255.2/32
set policy-options policy-statement adv_direct term 1 from protocol direct
set policy-options policy-statement adv_direct term 1 from route-filter 172.16.0.0/16 orlonger
set policy-options policy-statement adv_direct term 1 then community add map2bronze
set policy-options policy-statement adv_direct term 1 then accept
set policy-options community map2bronze members color:0:200
set policy-options community map2gold members color:0:100
set protocols bgp group PE2 type external
set protocols bgp group PE2 export adv_direct
set protocols bgp group PE2 peer-as 65412
set protocols bgp group PE2 neighbor 172.16.2.2
set routing-options router-id 172.16.255.2
set routing-options autonomous-system 64520
set system host-name CE2
```

PE1 (DUT)

```
set interfaces ge-0/0/0 unit 0 description "Link from PE1 to CE1"
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.2/30
set interfaces ge-0/0/1 unit 0 description "Link from PE1 to P1"
set interfaces ge-0/0/1 unit 0 family inet address 10.1.23.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description "Link from PE1 to P2"
set interfaces ge-0/0/2 unit 0 family inet address 10.1.24.1/24
```

```
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 unit 0 description "Link from PE1 to P3"
set interfaces ge-0/0/3 unit 0 family inet address 10.1.25.1/24
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set routing-instances CE1_L3vpn instance-type vrf
set routing-instances CE1_L3vpn protocols bgp group CE1 type external
set routing-instances CE1_L3vpn protocols bgp group CE1 peer-as 64510
set routing-instances CE1_L3vpn protocols bgp group CE1 neighbor 172.16.1.1
set routing-instances CE1_L3vpn interface ge-0/0/0.0
set routing-instances CE1_L3vpn route-distinguisher 192.168.0.1:12
set routing-instances CE1_L3vpn vrf-target target:65412:12
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.1
set protocols bgp group ibgp family inet unicast
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 192.168.0.2
set protocols mpls icmp-tunneling
set protocols mpls label-switched-path lsp_to_pe2 to 192.168.0.2
set protocols mpls label-switched-path lsp_to_pe2 primary best-effort
set protocols mpls label-switched-path gold_lsp_to_pe2 to 192.168.0.2
set protocols mpls label-switched-path gold_lsp_to_pe2 preference 5
set protocols mpls label-switched-path gold_lsp_to_pe2 primary gold
set protocols mpls label-switched-path gold_lsp_to_pe2 transport-class gold
set protocols mpls label-switched-path bronze_lsp_to_pe2 to 192.168.0.2
set protocols mpls label-switched-path bronze_lsp_to_pe2 preference 5
set protocols mpls label-switched-path bronze_lsp_to_pe2 primary bronze
set protocols mpls label-switched-path bronze_lsp_to_pe2 transport-class bronze
set protocols mpls path gold 10.1.23.2 strict
set protocols mpls path bronze 10.1.24.2 strict
set protocols mpls path best-effort 10.1.25.2 strict
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set protocols mpls interface ge-0/0/3.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface ge-0/0/2.0
set protocols rsvp interface ge-0/0/3.0
```

```
set routing-options route-distinguisher-id 192.168.0.1
set routing-options resolution preserve-nexthop-hierarchy
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 65412
set routing-options transport-class name gold color 100
set routing-options transport-class name bronze color 200
set system host-name PE1
```

PE2

```
set interfaces ge-0/0/0 unit 0 description "Link from PE2 to P1"
set interfaces ge-0/0/0 unit 0 family inet address 10.1.36.2/24
set interfaces ge-0/0/0 unit 0 family mpls

set interfaces ge-0/0/1 unit 0 description "Link from PE2 to P2"
set interfaces ge-0/0/1 unit 0 family inet address 10.1.46.2/24
set interfaces ge-0/0/1 unit 0 family mpls

set interfaces ge-0/0/2 unit 0 description "Link from PE2 to P3"
set interfaces ge-0/0/2 unit 0 family inet address 10.1.56.2/24
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 unit 0 description "Link from PE2 to CE2"
set interfaces ge-0/0/3 unit 0 family inet address 172.16.2.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set routing-instances CE2_L3vpn instance-type vrf
set routing-instances CE2_L3vpn protocols bgp group CE2 type external
set routing-instances CE2_L3vpn protocols bgp group CE2 peer-as 64520
set routing-instances CE2_L3vpn protocols bgp group CE2 neighbor 172.16.2.1
set routing-instances CE2_L3vpn interface ge-0/0/3.0
set routing-instances CE2_L3vpn route-distinguisher 192.168.0.2:12
set routing-instances CE2_L3vpn vrf-target target:65412:12
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.2
set protocols bgp group ibgp family inet unicast
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 192.168.0.1
set protocols mpls icmp-tunneling
set protocols mpls label-switched-path lsp_to_pe1 to 192.168.0.1
set protocols mpls label-switched-path gold_lsp_to_pe1 to 192.168.0.1
set protocols mpls label-switched-path gold_lsp_to_pe1 transport-class gold
set protocols mpls label-switched-path gold_lsp_to_pe1 preference 5
```

```
set protocols mpls label-switched-path bronze_lsp_to_pe1 to 192.168.0.1
set protocols mpls label-switched-path bronze_lsp_to_pe1 transport-class bronze
set protocols mpls label-switched-path bronze_lsp_to_pe1 preference 5
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface ge-0/0/2.0
set routing-options route-distinguisher-id 192.168.0.2
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 65412
set routing-options transport-class name gold color 100
set routing-options transport-class name bronze color 200
set system host-name PE2
```

P1

```
set interfaces ge-0/0/0 unit 0 description "Link from P1 to PE1"
set interfaces ge-0/0/0 unit 0 family inet address 10.1.23.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description "Link from P1 to PE2"
set interfaces ge-0/0/1 unit 0 family inet address 10.1.36.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.11/32
set protocols mpls icmp-tunneling
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface ge-0/0/1.0
```

```
set routing-options router-id 192.168.0.11
set system host-name P1
```

P2

```
set interfaces ge-0/0/0 unit 0 description "Link from P2 to PE1"
set interfaces ge-0/0/0 unit 0 family inet address 10.1.24.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description "Link from P2 to PE2"
set interfaces ge-0/0/1 unit 0 family inet address 10.1.46.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.12/32
set protocols mpls icmp-tunneling
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface ge-0/0/1.0
set routing-options router-id 192.168.0.12
set system host-name P2
```

P3

```
set interfaces ge-0/0/0 unit 0 description "Link from P3 to PE1"
set interfaces ge-0/0/0 unit 0 family inet address 10.1.25.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description "Link from P3 to PE2"
set interfaces ge-0/0/1 unit 0 family inet address 10.1.56.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.13/32
set protocols mpls icmp-tunneling
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
```

```
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface ge-0/0/1.0
set routing-options router-id 192.168.0.13
set system host-name P3
```

Appendix 3: Show Configuration Output on PE1

IN THIS SECTION

The Complete PE1 configuration in Curly Brace Format

```
user@PE1# show | no-more
system {
    host-name PE1;
}
interfaces {
    ge-0/0/0 {
        unit 0 {
            description "Link from PE1 to CE1";
            family inet {
                address 172.16.1.2/30;
            }
        }
    }
    ge-0/0/1 {
        unit 0 {
            description "Link from PE1 to P1";
            family inet {
                address 10.1.23.1/24;
            }
            family mpls;
        }
    }
    ge-0/0/2 {
```

```
        unit 0 {
            description "Link from PE1 to P2";
            family inet {
                address 10.1.24.1/24;
            }
            family mpls;
        }
    }
    ge-0/0/3 {
        unit 0 {
            description "Link from PE1 to P3";
            family inet {
                address 10.1.25.1/24;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.1/32;
            }
        }
    }
}
routing-instances {
    CE1_L3vpn {
        instance-type vrf;
        protocols {
            bgp {
                group CE1 {
                    type external;
                    peer-as 64510;
                    neighbor 172.16.1.1;
                }
            }
        }
        interface ge-0/0/0.0;
        route-distinguisher 192.168.0.1:12;
        vrf-target target:65412:12;
    }
}
routing-options {
```

```
    route-distinguisher-id 192.168.0.1;
    resolution {
        preserve-nexthop-hierarchy;
    }
    router-id 192.168.0.1;
    autonomous-system 65412;
    transport-class {
        name gold {
            color 100;
        }
        name bronze {
            color 200;
        }
    }
}
protocols {
    bgp {
        group ibgp {
            type internal;
            local-address 192.168.0.1;
            family inet {
                unicast;
            }
            family inet-vpn {
                unicast;
            }
            neighbor 192.168.0.2;
        }
    }
    mpls {
        label-switched-path lsp_to_pe2 {
            to 192.168.0.2;
            primary best-effort;
        }
        label-switched-path gold_lsp_to_pe2 {
            to 192.168.0.2;
            preference 5;
            primary gold;
            transport-class gold;
        }
        label-switched-path bronze_lsp_to_pe2 {
            to 192.168.0.2;
            preference 5;
```

```
            primary bronze;
            transport-class bronze;
        }
        path gold {
            10.1.23.2 strict;
        }
        path bronze {
            10.1.24.2 strict;
            10.1.66.6 strict;
        }
        path best-effort {
            10.1.25.2 strict;
        }
        icmp-tunneling;
        interface ge-0/0/1.0;
        interface ge-0/0/2.0;
        interface ge-0/0/3.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface lo0.0;
            interface ge-0/0/1.0;
            interface ge-0/0/2.0;
            interface ge-0/0/3.0;
        }
    }
    rsvp {
        interface lo0.0;
        interface ge-0/0/1.0;
        interface ge-0/0/2.0;
        interface ge-0/0/3.0;
    }
}
```

SEE ALSO

- BGP Classful Transport Demonstration in Juniper Vlabs
- IETF Specification: BGP Classful Transport Planes

Color-Based Mapping of VPN Services Overview

IN THIS SECTION

You can specify color as a protocol next hop constraint (in addition to the IPv4 or IPv6 address) for resolving transport tunnels over static colored and BGP segment routing traffic-engineered (SR-TE) LSPs. This is called the color-IP protocol next hop resolution, where you are required to configure a resolution-map and apply to the VPN services. With this feature, you can enable color-based traffic steering of Layer 2 and Layer 3 VPN services.

VPN Service Coloring

In general, a VPN service may be assigned a color on the egress router where the VPN NLRI is advertised, or on an ingress router where the VPN NLRI is received and processed.

You can assign a color to the VPN services at different levels:

- Per routing instance.
- Per BGP group.
- Per BGP neighbor.
- Per prefix.
- Set of prefixes.

Once you assign a color, the color is attached to a VPN service in the form of BGP color extended community.

You can assign multiple colors to a VPN service, referred to as multi-color VPN services. In such cases, the smallest color value is considered as the color of the VPN service, and all other colors are ignored.

Multiple colors are assigned by egress and/or ingress devices through multiple policies in the following order:

- BGP export policy on the egress device.
- BGP import policy on the ingress device.
- VRF import policy on the ingress device.

The two modes of VPN service coloring are:

Egress Color Assignment

In this mode, the egress device (that is, the advertiser of the VPN NLRI) is responsible for coloring the VPN service. To enable this mode, you can define a routing policy, and apply it in the VPN service's routing-instance vrf-export, group export, or group neighbor export at the [edit protocols bgp] hierarchy level. The VPN NLRI is advertised by BGP with the specified color extended community.

For example:

```
[edit policy-options]
community red-comm {
  members color:0:50;
}
```

```
[edit policy-options]
policy-statement pol-color {
  term t1 {
    from {
     [any match conditions];
    }
    then {
     community add red-comm;
     accept;
    }
  }
}
```

```
[edit routing-instances]
vpn-X {
...
```

```
  vrf-export pol-color ...;
}
```

OR

NOTE: When you apply the routing policy as an export policy of a BGP group or BGP neighbor, you must include the `vpn-apply-export` statement at the BGP, BGP group, or BGP neighbor level in order for the policy to take an effect on the VPN NLRI.

```
[edit protocols bgp]
group PEs {
...
  neighbor PE-A {
  export pol-color ...;
  vpn-apply-export;
  }
}
```

The routing policies are applied to Layer 3 VPN prefix NLRIs, Layer 2 VPN NRLIs, and EVPN NLRIs. The color extended community is inherited by all the VPN routes, imported, and installed in the target VRFs on one or multiple ingress devices.

Ingress Color Assignment

In this mode, the ingress device (that is, the receiver of the VPN NLRI) is responsible for coloring the VPN service. To enable this mode, you can define a routing policy, and apply it to the VPN service's routing-instance `vrf-import`, group import, or group neighbor import at the `[edit protocols bgp]` hierarchy level. All the VPN routes matching the routing policy is attached with the specified color extended community.

For example:

```
[edit policy-options]
community red-comm {
  members color:0:50;
}
```

```
[edit policy-options]
policy-statement pol-color {
```

```
  term t1 {
    from {
    [any match conditions];
}
then {
  community add red-comm;
    accept;
    }
  }
}
```

```
[edit routing-instances]
vpn-Y {
...
  vrf-import pol-color ...;
}
```

OR

```
[edit protocols bgp]
group PEs {
...
  neighbor PE-B {
    import pol-color ...;
  }
}
```

Specifying VPN Service Mapping Mode

To specify flexible VPN service mapping modes, you must define a policy using the resolution-map statement, and refer the policy in a VPN service's routing-instance vrf-import, group import, or group neighbor import at the [edit protocols bgp] hierarchy level. All the VPN routes matching the routing policy are attached with the specified resolution-map.

For example:

```
[edit policy-options]
resolution-map map-A {
  <mode-1>;
  <mode-2>;
```

```
  ...
}
policy-statement pol-resolution {
  term t1 {
    from {
    [any match conditions];
}
then {
  resolution-map map-A;
    accept;
    }
  }
}
```

You can apply import policy to the VPN service's routing-instance.

```
[edit routing-instances]
vpn-Y {
...
  vrf-import pol-resolution ...;
}
```

You can also apply the import policy to a BGP group or BGP neighbor.

```
[edit protocols bgp]
group PEs {
...
  neighbor PE-B {
  import pol-resolution ...;
  }
}
```

NOTE: Each VPN service mapping mode should have a unique name defined in the resolution-map. Only a single entry of IP-color is supported in the resolution-map, where the VPN route(s) are resolved using a colored-IP protocol next hop in the form of ip-address:color over the inetcolor.0 and inet6color.0 routing tables.

Color-IP Protocol Next Hop Resolution

The protocol next hop resolution process is enhanced to support colored-IP protocol next hop resolution. For a colored VPN service, the protocol next hop resolution process takes a color and a resolution-map, builds a colored-IP protocol next hop in the form of `ip-address:color`, and resolves the protocol next hop in the inetcolor.0 and inet6color.0 routing tables.

You must configure a policy to support multipath resolution of colored Layer 2 VPN, Layer 3 VPN, or EVPN services over colored LSPs. The policy must then be applied with the relevant RIB table as the resolver import policy.

For example:

```
[edit policy-options]
policy-statement mpath {
  then multipath-resolve;
}
```

```
[edit routing-options]
resolution {
  rib bgp.l3vpn.0 {
    inetcolor-import mpath;
  }
}

resolution {
  rib bgp.l3vpn-inet6.0 {
    inet6color-import mpath;
  }
}

resolution {
  rib bgp.l2vpn.0 {
    inetcolor-import mpath;
  }
}

resolution {
  rib mpls.0 {
    inetcolor-import mpath;
  }
}
```

```
resolution {
  rib bgp.evpn.0 {
    inetcolor-import mpath;
  }
}
```

Fallback to IP Protocol Next Hop Resolution

If a colored VPN service does not have a resolution-map applied to it, the VPN service ignores its color and falls back to the IP protocol next hop resolution. Conversely, if a non-colored VPN service has a resolution-map applied to it, the resolution-map is ignored, and the VPN service uses the IP protocol next hop resolution.

The fallback is a simple process from colored SR-TE LSPs to LDP LSPs, by using a RIB group for LDP to install routes in inet{6}color.0 routing tables. A longest prefix match for a colored-IP protocol next hop ensures that if a colored SR-TE LSP route does not exist, an LDP route with a matching IP address should be returned.

BGP Labeled Unicast Color-based Mapping over SR-TE and IS-IS Underlay

BGP Labeled Unicast (BGP-LU) can resolve IPv4 or IPv6 routes over segment routing–traffic engineering (SR-TE) with IS-IS underlay for both IPv4 and IPv6 address families. BGP-LU supports mapping a BGP community color and defining a `resolution map` for SR-TE. A colored protocol next hop is constructed and it is resolved on a colored SR-TE tunnel in the `inetcolor.0` or `inet6color.0` table. Thus BGP-LU resolves protocol next hop over SR-TE tunnels for packet transport. BGP uses `inet.3` and `inet6.3` tables for non-color based mapping.

Supported and Unsupported Features for Color-Based Mapping of VPN Services

The following features and functionality are supported with color-based mapping of VPN services:

- BGP Layer 3 VPN
- BGP Layer 2 VPN (Kompella Layer 2 VPN)
- BGP EVPN
- Resolution-map with a single IP-color option.
- Colored IPv4 and IPv6 protocol next hop resolution.
- Routing information base (also known as routing table) group based fallback to LDP LSP in inetcolor.0 or inet6color.0 routing tables.

- Colored SR-TE LSP.
- Virtual platforms.
- 64-bit Junos OS.
- Logical systems.
- BGP Labeled Unicast

The following features and functionality are not supported with color-based mapping of VPN services:

- Colored MPLS LSPs, such as RSVP, LDP, BGP-LU, static.
- Layer 2 circuit
- FEC-129 BGP auto-discovered and LDP-signaled Layer 2 VPN.
- VPLS
- MVPN
- IPv4 and IPv6 using resolution-map.

SEE ALSO

> Understanding Static Segment Routing LSP in MPLS Networks
>
> resolution-map

RELATED DOCUMENTATION

> *Basic MPLS Configuration*

Color-Based Mapping of VPN Services for SR-MPLS Segment Routing LSPs

SUMMARY

IN THIS SECTION

Color-Based Mapping of VPN Services

You can specify color as a protocol next hop constraint (in addition to the IPv4 or IPv6 address) for resolving transport tunnels over static colored and BGP segment routing traffic-engineered (SR-TE) LSPs. This is called the color-IP protocol next hop resolution, where you are required to configure a resolution-map and apply to the VPN services. With this feature, you can enable color-based traffic steering of Layer 2 and Layer 3 VPN services.

Junos OS supports colored SR-TE LSPs associated with a single color. The color-based mapping of VPN services feature is supported on static colored LSPs and BGP SR-TE LSPs.

VPN Service Coloring

In general, a VPN service may be assigned a color on the egress router where the VPN NLRI is advertised, or on an ingress router where the VPN NLRI is received and processed.

You can assign a color to the VPN services at different levels:

- Per routing instance.
- Per BGP group.
- Per BGP neighbor.
- Per prefix.

Once you assign a color, the color is attached to a VPN service in the form of BGP color extended community.

You can assign multiple colors to a VPN service, referred to as multi-color VPN services. In such cases, the last color attached is considered as the color of the VPN service, and all other colors are ignored.

Multiple colors are assigned by egress and/or ingress devices through multiple policies in the following order:

- BGP export policy on the egress device.
- BGP import policy on the ingress device.
- VRF import policy on the ingress device.

The two modes of VPN service coloring are:

Egress Color Assignment

In this mode, the egress device (that is, the advertiser of the VPN NLRI) is responsible for coloring the VPN service. To enable this mode, you can define a routing policy, and apply it in the VPN service's routing-instance `vrf-export`, group export, or group neighbor export at the `[edit protocols bgp]` hierarchy level. The VPN NLRI is advertised by BGP with the specified color extended community.

For example:

```
[edit policy-options]
community red-comm {
    members color:0:50;
}
```

```
[edit policy-options]
policy-statement pol-color {
    term t1 {
        from {
            [any match conditions];
        }
        then {
            community add red-comm;
            accept;
        }
    }
}
```

```
[edit routing-instances]
vpn-X {
    ...
    vrf-export pol-color ...;
}
```

Or

NOTE: When you apply the routing policy as an export policy of a BGP group or BGP neighbor, you must include the `vpn-apply-export` statement at the BGP, BGP group, or BGP neighbor level in order for the policy to take an effect on the VPN NLRI.

```
[edit protocols bgp]
group PEs {
    ...
    neighbor PE-A {
        export pol-color ...;
        vpn-apply-export;
    }
}
```

The routing policies are applied to Layer 3 VPN prefix NLRIs, Layer 2 VPN NRLIs, and EVPN NLRIs. The color extended community is inherited by all the VPN routes, imported, and installed in the target VRFs on one or multiple ingress devices.

Ingress Color Assignment

In this mode, the ingress device (that is, the receiver of the VPN NLRI) is responsible for coloring the VPN service. To enable this mode, you can define a routing policy, and apply it to the VPN service's routing-instance `vrf-import`, group import, or group neighbor import at the `[edit protocols bgp]` hierarchy level. All the VPN routes matching the routing policy is attached with the specified color extended community.

For example:

```
[edit routing-options]
community red-comm {
    members color:0:50;
}
```

```
[edit policy-options]
policy-statement pol-color {
    term t1 {
        from {
            [any match conditions];
```

```
        }
        then {
            community add red-comm;
            accept;
        }
    }
}
```

```
[edit routing-instances]
vpn-Y {
    ...
    vrf-import pol-color ...;
}
```

Or

```
[edit protocols bgp]
group PEs {
    ...
    neighbor PE-B {
        import pol-color ...;
    }
}
```

Specifying VPN Service Mapping Mode

To specify flexible VPN service mapping modes, you must define a policy using the `resolution-map` statement, and refer the policy in a VPN service's routing-instance `vrf-import`, group import, or group neighbor import at the `[edit protocols bgp]` hierarchy level. All the VPN routes matching the routing policy are attached with the specified resolution-map.

For example:

```
[edit policy-options]
resolution-map map-A {
    <mode-1>;
    <mode-2>;
    ...
}
```

```
policy-statement pol-resolution {
    term t1 {
        from {
            [any match conditions];
        }
        then {
            resolution-map map-A;
            accept;
        }
    }
}
```

You can apply import policy to the VPN service's routing-instance.

```
[edit routing-instances]
vpn-Y {
    ...
    vrf-import pol-resolution ...;
}
```

You can also apply the import policy to a BGP group or BGP neighbor.

```
[edit protocols bgp]
group PEs {
    ...
    neighbor PE-B {
        import pol-resolution ...;
    }
}
```

NOTE: Each VPN service mapping mode should have a unique name defined in the resolution-map. Only a single entry of IP-color is supported in the resolution-map, where the VPN route(s) are resolved using a colored-IP protocol next hop in the form of `ip-address:color`.

Color-IP Protocol Next Hop Resolution

The protocol next hop resolution process is enhanced to support colored-IP protocol next hop resolution. For a colored VPN service, the protocol next hop resolution process takes a color and a

resolution-map, builds a colored-IP protocol next hop in the form of *IP-address:color*, and resolves the protocol next hop in the inet6color.0 routing table.

You must configure a policy to support multipath resolution of colored Layer 2 VPN, Layer 3 VPN, or EVPN services over colored LSPs. The policy must then be applied with the relevant RIB table as the resolver import policy.

For example:

```
[edit policy-options]
policy-statement mpath {
    then multipath-resolve;
}
```

```
[edit routing-options]
resolution {
    rib bgp.l3vpn.0 {
        inetcolor-import mpath;
    }
}

resolution {
    rib bgp.l3vpn-inet6.0 {
        inet6color-import mpath;
    }
}

resolution {
    rib bgp.l2vpn.0 {
        inetcolor-import mpath;
    }
}

resolution {
    rib mpls.0 {
        inetcolor-import mpath;
    }
}

resolution {
    rib bgp.evpn.0 {
        inetcolor-import mpath;
```

```
    }
}
```

Fallback to IP Protocol Next Hop Resolution

If a colored VPN service does not have a resolution-map applied to it, the VPN service ignores its color and falls back to the IP protocol next hop resolution. Conversely, if a non-colored VPN service has a resolution-map applied to it, the resolution-map is ignored, and the VPN service uses the IP protocol next hop resolution.

The fallback is a simple process from colored SR-TE LSPs to LDP LSPs, by using a RIB group for LDP to install routes in inet{6}color.0 routing tables. A longest prefix match for a colored-IP protocol next hop ensures that if a colored SR-TE LSP route does not exist, an LDP route with a matching IP address should be returned.

BGP Labeled Unicast Color-based Mapping over SR-TE

BGP Labeled Unicast (BGP-LU) can resolve IPv4 or IPv6 routes over segment routing–traffic engineering (SR-TE) for both IPv4 and IPv6 address families. BGP-LU supports mapping a BGP community color and defining a `resolution map` for SR-TE. A colored protocol next hop is constructed and it is resolved on a colored SR-TE tunnel in the `inetcolor.0` or `inet6color.0` table. BGP uses `inet.3` and `inet6.3` tables for non-color based mapping. This enables you to advertise BGP-LU IPv6 and IPv4 prefixes with an IPv6 next-hop address in IPv6-only networks where routers do not have any IPv4 addresses configured. With this feature, Currently we support BGP IPv6 LU over SR-TE with IS-IS underlay.

NOTE: Documentation updates with a detailed description of BGP Labeled Unicast-based mapping over SR-TE will be available in upcoming revisions.

BGP-LU supports the following scenarios:

- BGP IPv4 LU over colored BGP IPv4 SR-TE, with ISIS/OSPF IPv4 SR extensions.
- BGP IPv4 LU over static colored and non-colored IPv4 SR-TE, with ISIS/OSPF IPv4 SR extensions.
- BGP IPv6 LU over colored BGP IPv6 SR-TE, with ISIS IPv6 SR extensions.
- BGP IPv6 LU over static colored and non-colored IPv6 SR-TE, with ISIS IPv6 SR extensions.
- IPv6 Layer 3 VPN services with IPv6 local address and IPv6 neighbor address.
- IPv6 Layer 3 VPN services over BGP IPv6 SR-TE, with ISIS IPv6 SR extensions.
- IPv6 Layer 3 VPN services over static-colored and non-colored IPv6 SR-TE, with ISIS IPv6 SR extensions.

Supported and Unsupported Features for Color-Based Mapping of VPN Services

The following features and functionality are supported with color-based mapping of VPN services:

- BGP Layer 2 VPN (Kompella Layer 2 VPN)
- BGP EVPN
- Resolution-map with a single IP-color option.
- Colored IPv4 and IPv6 protocol next hop resolution.
- Routing information base (also known as routing table) group based fallback to LDP LSP in inetcolor.0 routing table.
- Colored SR-TE LSP.
- Virtual platforms.
- 64-bit Junos OS.
- Logical systems.
- BGP labeled unicast.

The following features and functionality are not supported with color-based mapping of VPN services:

- Colored MPLS LSPs, such as RSVP, LDP, BGP-LU, static.
- Layer 2 circuit
- FEC-129 BGP auto-discovered and LDP-signaled Layer 2 VPN.
- VPLS
- MVPN
- IPv4 and IPv6 using resolution-map.

Interdomain Segment Routing

IN THIS SECTION

How to Configure Multiple Independent IGP Instances of IS-IS and OSPFv2

SUMMARY

Learn how to configure and run multiple instances of IGP on a router.

IN THIS SECTION

Configure Multiple IGP Instances of IS-IS

SUMMARY

Learn about the benefits and get an overview of running multiple interior gateway protocol (IGP) instances of IS-IS on a router.

IN THIS SECTION

Benefits of Multi-Instance IS-IS

- You can use multiple IGP instances of IS-IS to redistribute routes among independent IS-IS domains on a single router.
- You can construct flexible IS-IS hierarchies across independent IGP domains.
- Allows decoupling of multiple IS-IS flooding domains and therefore achieve a more scalable IS-IS deployment.

Figure 26: Multi-Instance IS-IS Deployment Topology

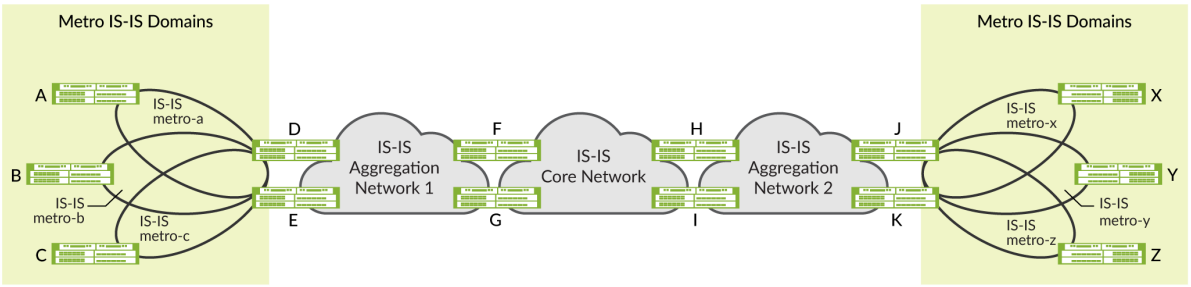


Figure 26 on page 344 illustrates several benefits of configuring multiple IGP instances of IS-IS on the router. For example, Router F participates in two independent IS-IS instances. Router F treats IS-IS Aggregation Network-1 and IS-IS Core Network as two independent IGP domains, while at the same time redistributing routes between those domains. Network operators can use this flexibility to construct a hierarchy of IS-IS domains.

Figure 26 on page 344 also illustrates the use of multiple IGP instances of IS-IS to separate metro networks into independent IS-IS flooding domains. In the example, routers D and E participate in the IS-IS metro-a, IS-IS metro-b, and IS-IS metro-c networks, as well as in IS-IS Aggregation Network-1. Routers D and E do not flood the different IS-IS domains with IS-IS advertisements. Instead they redistribute specific routes among the different IS-IS domains, which allows for more scalable metro deployments.

Multi-Instance IS-IS Overview

You can configure and run multiple independent IGP instances of IS-IS simultaneously on a router. These instances are associated with the default routing instance, and they install routes in the default routing table.Each IS-IS instance can also export the routes installed in the routing table by other IS-IS instances using the standard Junos OS routing policy configuration. By default, the routes installed by the different IS-IS instances have the same route preference.

NOTE: Junos OS does not support configuring the same logical interface in multiple IGP instances of IS-IS.

In most deployment scenarios, only one IS-IS instance on a router installs a route for a given prefix. Therefore, you don't need to configure different route preferences for multiple IS-IS instances. However, for certain deployment scenarios where multiple IS-IS instances install the routes for the same prefix in the routing table, you can set a different route preference for the routes installed by other IS-IS instances. This allows the routing table to choose the routes with the best route preference and installs those routes in the forwarding table.

You can use the multiple IS-IS instance feature for both hierarchical and parallel deployments. In the case of hierarchical deployments, there are well-defined borders between the groups of routers participating in different IGP instances. In parallel deployments, different IGP instances (typically not more than two or three) span entire groups of routers. You can also have mixed deployments, with some domains in a hierarchical deployment running IGP instances in parallel.

You can configure multiple independent IGP instances of IS-IS by including the `isis-instance` configuration statement at the [`edit protocols`] hierarchy level. The configuration statements that you use at the [`edit protocols isis-instance` *`igp-instance-name`*] hierarchy level are the same as those available at the [`edit protocols isis`] hierarchy level.

NOTE: The `isis-instance` configuration statement is not supported at the [`edit routing-instances` *`routing-instance-name`* `protocols`] hierarchy level.

You can configure multiple independent IGP instances of OSPFv2 by including the `ospf-instance` configuration statement at the [`edit protocols`] hierarchy level. The configuration statements that you use at the [`edit protocols ospf-instance` *`igp-instance-name`*] hierarchy level are the same as those available at the [`edit protocols ospf`] hierarchy level.

NOTE: The `ospf-instance` configuration statement is not supported at the [`edit routing-instances` *`routing-instance-name`* `protocols`] hierarchy level.

You can configure and run multiple independent interior gateway protocol (IGP) instances of OSPFv2 with segment routing (SR) on a router. You can create two or more OSPF instances and apply SR-MPLS on each instance. Multiple instances of OSPF can advertise different prefix-segment identifiers (prefix-SIDs). Other instances can use these SIDs for making routing decisions.

Multi-instance OSPF combined with SR enhances network flexibility, scalability, and control over traffic engineering, especially in large and complex networks.

Example: Configure Independent IS-IS Instances in Metro Flooding Domains

SUMMARY

Use this example to learn how to configure independent metro flooding domains running multiple IGP instances of IS-IS.

IN THIS SECTION

Overview

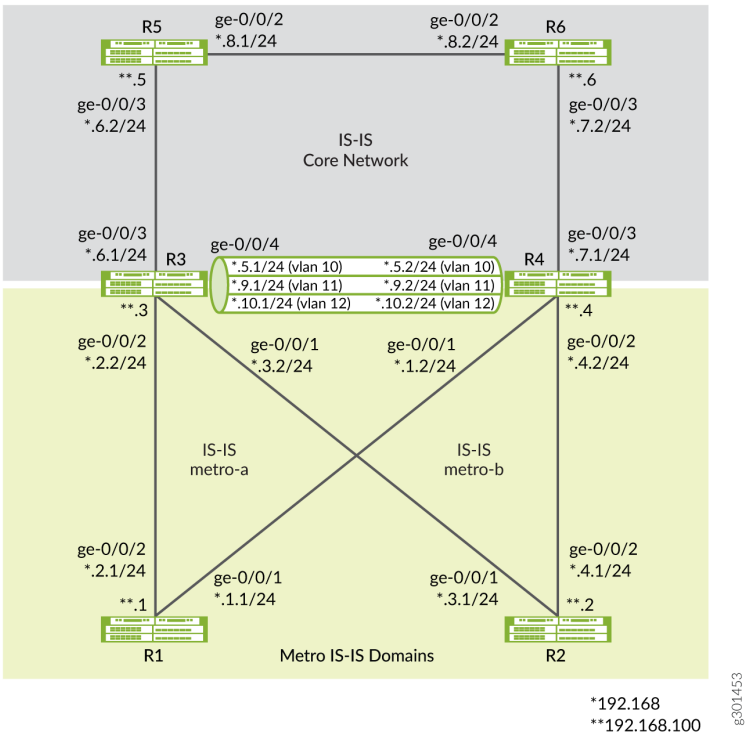
IN THIS SECTION

This example shows how to configure and run multiple independent IGP instances of IS-IS in metro flooding domains.

Topology

Figure 2 shows an example of metro flooding domains (metro-a and metro-b) running independent IGP instances of IS-IS. In the topology, routers R3 and R4 participate in metro IS-IS domains (IS-IS metro-a and IS-IS metro-b) and the IS-IS core network domain. Routers R3 and R4 do not flood the different IS-IS domains with IS-IS advertisements. Instead they redistribute specific routes among the different IS-IS domains, which allows for a more scalable metro deployment.

Figure 27: Multi-Instance IS-IS Topology Across Independent Metro Flooding Domains (IGP Domains)



Requirements

This example uses the following hardware and software components:

- MX Series routers
- Junos OS Release 21.1R1 or later running on all devices

NOTE: You must configure the network services mode as Enhanced IP. The Enhanced IP configuration ensures that the router uses enhanced mode capabilities.

```
[edit]
user@CE1#set chassis network-services enhanced-ip
```

After you configure the `enhanced-ip` statement and commit the configuration, the following warning message appears, prompting you to reboot the router:

```
'chassis'
WARNING: Chassis configuration for network services has been changed. A system reboot
```

```
is mandatory. Please reboot the system NOW. Continuing without a reboot might result
in unexpected system behavior.
commit complete
```

The reboot brings up the FPCs on the router.

[See show chassis network-services.]

Configuration

IN THIS SECTION

To configure and run multiple IGP instances of IS-IS on the router, perform these tasks:

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Device R1

```
set interfaces ge-0/0/1 description R1-to-R4
set interfaces ge-0/0/1 unit 0 family inet address 192.168.1.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/2 description R1-to-R3
set interfaces ge-0/0/2 unit 0 family inet address 192.168.2.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.100.1/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0001.00
set protocols isis interface ge-0/0/1.0 level 2 metric 100
set protocols isis interface ge-0/0/1.0 level 1 disable
set protocols isis interface ge-0/0/1.0 point-to-point
```

```
set protocols isis interface ge-0/0/2.0 level 1 disable
set protocols isis interface ge-0/0/2.0 level 2 metric 100
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface lo0.0 passive
set routing-options router-id 192.168.100.1
```

Device R2

```
set interfaces ge-0/0/1 description R2-to-R3
set interfaces ge-0/0/1 unit 0 family inet address 192.168.3.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/2 description R2-to-R4
set interfaces ge-0/0/2 unit 0 family inet address 192.168.4.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.100.2/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0002.00
set protocols isis interface ge-0/0/1.0 level 1 disable
set protocols isis interface ge-0/0/1.0 level 2 metric 100
set protocols isis interface ge-0/0/1.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 1 disable
set protocols isis interface ge-0/0/2.0 level 2 metric 100
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface lo0.0 passive
set routing-options router-id 192.168.100.2
```

Device R3

```
set interfaces ge-0/0/1 description R3-to-R2
set interfaces ge-0/0/1 unit 0 family inet address 192.168.3.2/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/2 description R3-to-R1
set interfaces ge-0/0/2 unit 0 family inet address 192.168.2.2/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/3 description R3-to-R5
set interfaces ge-0/0/3 unit 0 family inet address 192.168.6.1/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/4 description R3-to-R4
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 10
```

```
set interfaces ge-0/0/4 unit 0 family inet address 192.168.5.1/24
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 1 vlan-id 11
set interfaces ge-0/0/4 unit 1 family inet address 192.168.9.1/24
set interfaces ge-0/0/4 unit 1 family iso
set interfaces ge-0/0/4 unit 2 vlan-id 12
set interfaces ge-0/0/4 unit 2 family inet address 192.168.10.1/24
set interfaces ge-0/0/4 unit 2 family iso
set interfaces lo0 unit 0 family inet address 192.168.100.3/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0003.00
set policy-options policy-statement export-direct-loopback from protocol direct
set policy-options policy-statement export-direct-loopback from route-filter 192.168.100.3/32
exact
set policy-options policy-statement export-direct-loopback then accept
set policy-options policy-statement export-isis from protocol isis
set policy-options policy-statement export-isis from level 2
set policy-options policy-statement export-isis from route-filter 192.168.100.0/24 longer
set policy-options policy-statement export-isis then accept
set policy-options policy-statement export-isis-metro-a from igp-instance metro-a
set policy-options policy-statement export-isis-metro-a from protocol isis
set policy-options policy-statement export-isis-metro-a from level 2
set policy-options policy-statement export-isis-metro-a from route-filter 192.168.100.0/24 longer
set policy-options policy-statement export-isis-metro-a then accept
set policy-options policy-statement export-isis-metro-b from igp-instance metro-b
set policy-options policy-statement export-isis-metro-b from protocol isis
set policy-options policy-statement export-isis-metro-b from level 2
set policy-options policy-statement export-isis-metro-b from route-filter 192.168.100.0/24 longer
set policy-options policy-statement export-isis-metro-b then accept
set protocols isis interface ge-0/0/3.0 level 1 disable
set protocols isis interface ge-0/0/3.0 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 level 1 disable
set protocols isis interface ge-0/0/4.0 level 2 metric 100
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis export export-isis-metro-a
set protocols isis export export-isis-metro-b
set protocols isis-instance metro-b interface ge-0/0/1.0 level 1 disable
set protocols isis-instance metro-b interface ge-0/0/1.0 level 2 metric 100
set protocols isis-instance metro-b interface ge-0/0/1.0 point-to-point
set protocols isis-instance metro-b interface ge-0/0/4.2 level 1 disable
set protocols isis-instance metro-b interface ge-0/0/4.2 level 2 metric 100
set protocols isis-instance metro-b interface ge-0/0/4.2 point-to-point
```

```
set protocols isis-instance metro-b export export-isis
set protocols isis-instance metro-b export export-direct-loopback
set protocols isis-instance metro-b export export-isis-metro-a
set protocols isis-instance metro-a interface ge-0/0/2.0 level 1 disable
set protocols isis-instance metro-a interface ge-0/0/2.0 level 2 metric 100
set protocols isis-instance metro-a interface ge-0/0/2.0 point-to-point
set protocols isis-instance metro-a interface ge-0/0/4.1 level 1 disable
set protocols isis-instance metro-a interface ge-0/0/4.1 level 2 metric 100
set protocols isis-instance metro-a interface ge-0/0/4.1 point-to-point
set protocols isis-instance metro-a export export-isis
set protocols isis-instance metro-a export export-direct-loopback
set protocols isis-instance metro-a export export-isis-metro-b
set routing-options router-id 192.168.100.3
```

Device R4

```
set interfaces ge-0/0/1 description R4-to-R1
set interfaces ge-0/0/1 unit 0 family inet address 192.168.1.2/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/2 description R4-to-R2
set interfaces ge-0/0/2 unit 0 family inet address 192.168.4.2/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/3 description R4-to-R6
set interfaces ge-0/0/3 unit 0 family inet address 192.168.7.1/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/4 description R4-to-R3
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 10
set interfaces ge-0/0/4 unit 0 family inet address 192.168.5.2/24
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 1 vlan-id 11
set interfaces ge-0/0/4 unit 1 family inet address 192.168.9.2/24
set interfaces ge-0/0/4 unit 1 family iso
set interfaces ge-0/0/4 unit 2 vlan-id 12
set interfaces ge-0/0/4 unit 2 family inet address 192.168.10.2/24
set interfaces ge-0/0/4 unit 2 family iso
set interfaces lo0 unit 0 family inet address 192.168.100.4/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0004.00
set policy-options policy-statement export-direct-loopback from protocol direct
set policy-options policy-statement export-direct-loopback from route-filter 192.168.100.4/32
exact
```

```
set policy-options policy-statement export-direct-loopback then accept
set policy-options policy-statement export-isis from protocol isis
set policy-options policy-statement export-isis from level 2
set policy-options policy-statement export-isis from route-filter 192.168.100.0/24 longer
set policy-options policy-statement export-isis then accept
set policy-options policy-statement export-isis-metro-a from igp-instance metro-a
set policy-options policy-statement export-isis-metro-a from protocol isis
set policy-options policy-statement export-isis-metro-a from level 2
set policy-options policy-statement export-isis-metro-a from route-filter 192.168.100.0/24 longer
set policy-options policy-statement export-isis-metro-a then accept
set policy-options policy-statement export-isis-metro-b from igp-instance metro-b
set policy-options policy-statement export-isis-metro-b from protocol isis
set policy-options policy-statement export-isis-metro-b from level 2
set policy-options policy-statement export-isis-metro-b from route-filter 192.168.100.0/24 longer
set policy-options policy-statement export-isis-metro-b then accept
set protocols isis interface ge-0/0/3.0 level 1 disable
set protocols isis interface ge-0/0/3.0 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 level 1 disable
set protocols isis interface ge-0/0/4.0 level 2 metric 100
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis export export-isis-metro-a
set protocols isis export export-isis-metro-b
set protocols isis-instance metro-a interface ge-0/0/1.0 level 1 disable
set protocols isis-instance metro-a interface ge-0/0/1.0 level 2 metric 100
set protocols isis-instance metro-a interface ge-0/0/1.0 point-to-point
set protocols isis-instance metro-a interface ge-0/0/4.1 level 1 disable
set protocols isis-instance metro-a interface ge-0/0/4.1 level 2 metric 100
set protocols isis-instance metro-a interface ge-0/0/4.1 point-to-point
set protocols isis-instance metro-a export export-isis
set protocols isis-instance metro-a export export-direct-loopback
set protocols isis-instance metro-a export export-isis-metro-b
set protocols isis-instance metro-b interface ge-0/0/2.0 level 1 disable
set protocols isis-instance metro-b interface ge-0/0/2.0 level 2 metric 100
set protocols isis-instance metro-b interface ge-0/0/2.0 point-to-point
set protocols isis-instance metro-b interface ge-0/0/4.2 level 1 disable
set protocols isis-instance metro-b interface ge-0/0/4.2 level 2 metric 100
set protocols isis-instance metro-b interface ge-0/0/4.2 point-to-point
set protocols isis-instance metro-b export export-isis
set protocols isis-instance metro-b export export-direct-loopback
```

```
set protocols isis-instance metro-b export export-isis-metro-a
set routing-options router-id 192.168.100.4
```

Device R5

```
set interfaces ge-0/0/2 description R5-to-R6
set interfaces ge-0/0/2 unit 0 family inet address 192.168.8.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/3 description R5-to-R3
set interfaces ge-0/0/3 unit 0 family inet address 192.168.6.2/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.100.5/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0005.00
set protocols isis interface ge-0/0/2.0 level 1 disable
set protocols isis interface ge-0/0/2.0 level 2 metric 100
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 1 disable
set protocols isis interface ge-0/0/3.0 level 2 metric 100
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface lo0.0 passive
set routing-options router-id 192.168.100.5
```

Device R6

```
set interfaces ge-0/0/2 description R6-to-R5
set interfaces ge-0/0/2 unit 0 family inet address 192.168.8.2/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/3 description R6-to-R4
set interfaces ge-0/0/3 unit 0 family inet address 192.168.7.2/24
set interfaces ge-0/0/3 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.100.6/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0006.00
set protocols isis interface ge-0/0/2.0 level 1 disable
set protocols isis interface ge-0/0/2.0 level 2 metric 100
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/3.0 level 1 disable
set protocols isis interface ge-0/0/3.0 level 2 metric 100
set protocols isis interface ge-0/0/3.0 point-to-point
```

```
set protocols isis interface lo0.0 passive
set routing-options router-id 192.168.100.6
```

Configure R1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see Using the CLI Editor in Configuration Mode in the CLI User Guide.

You can use the steps in this example to also configure the R2, R5, and R6 routers. See "CLI Quick Configuration" on page 348 and Figure 2 to understand the interface IDs, IP addresses, and the loopback addresses used on these routers.

To configure R1:

1. Configure the interfaces to enable IP (`inet`) and ISO family support.

```
user@R1# set interfaces ge-0/0/1 description R1-to-R4
user@R1# set interfaces ge-0/0/1 unit 0 family inet address 192.168.1.1/24
user@R1# set interfaces ge-0/0/1 unit 0 family iso
user@R1# set interfaces ge-0/0/2 description R1-to-R3
user@R1# set interfaces ge-0/0/2 unit 0 family inet address 192.168.2.1/24
user@R1# set interfaces ge-0/0/2 unit 0 family iso
```

2. Create the loopback interface and configure the IP and NET addresses.

```
user@R1# set interfaces lo0 unit 0 family inet address 192.168.100.1/32
user@R1# set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0001.00
```

3. Configure routing options to identify the router in the domain.

```
user@R1# set routing-options router-id 192.168.100.1
```

4. Enable IS-IS on the interfaces.

```
user@R1# set protocols isis interface ge-0/0/1.0 level 2 metric 100
user@R1# set protocols isis interface ge-0/0/1.0 level 1 disable
user@R1# set protocols isis interface ge-0/0/1.0 point-to-point
user@R1# set protocols isis interface ge-0/0/2.0 level 1 disable
user@R1# set protocols isis interface ge-0/0/2.0 level 2 metric 100
user@R1# set protocols isis interface ge-0/0/2.0 point-to-point
user@R1# set protocols isis interface lo0.0 passive
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, and `show protocols` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
interfaces {
    ge-0/0/1 {
        description R1-to-R4;
        unit 0 {
            family inet {
                address 192.168.1.1/24;
            }
            family iso;
        }
    }
    ge-0/0/2 {
        description R1-to-R3;
        unit 0 {
            family inet {
                address 192.168.2.1/24;
            }
            family iso;
        }
    }
    lo0 {
        unit 0 {
            family inet {
```

```
                address 192.168.100.1/32;
            }
            family iso {
                address 49.0002.0192.0168.0001.00;
            }
        }
    }
}
protocols {
    isis {
        interface ge-0/0/1.0 {
            level 2 metric 100;
            level 1 disable;
            point-to-point;
        }
        interface ge-0/0/2.0 {
            level 1 disable;
            level 2 metric 100;
            point-to-point;
        }
        interface lo0.0 {
            passive;
        }
    }
}
routing-options {
    router-id 192.168.100.1;
}
```

Configure R3

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see Using the CLI Editor in Configuration Mode in the CLI User Guide.

You can use the steps in this example to also configure the R4 router. See "CLI Quick Configuration" on page 348 and Figure 2 to understand the interface IDs, IP addresses, and the loopback address used on the router.

To configure R3:

1. Configure the interfaces connecting to R1, R2, and R5 to enable IP and ISO family support.

```
user@R3# set interfaces ge-0/0/1 description R3-to-R2
user@R3# set interfaces ge-0/0/1 unit 0 family inet address 192.168.3.2/24
user@R3# set interfaces ge-0/0/1 unit 0 family iso
user@R3# set interfaces ge-0/0/2 description R3-to-R1
user@R3# set interfaces ge-0/0/2 unit 0 family inet address 192.168.2.2/24
user@R3# set interfaces ge-0/0/2 unit 0 family iso
user@R3# set interfaces ge-0/0/3 description R3-to-R5
user@R3# set interfaces ge-0/0/3 unit 0 family inet address 192.168.6.1/24
user@R3# set interfaces ge-0/0/3 unit 0 family iso
```

2. Configure three subinterfaces (logical interfaces) connecting R3 and R4 (one IS-IS standard instance and two IS-IS metro instances (IS-IS metro-a and IS-IS metro-b)).

NOTE: The standard IS-IS instance refers to the IS-IS IGP instance configured at the [`edit protocols isis`] hierarchy level.

```
user@R3# set interfaces ge-0/0/4 description R3-to-R4
user@R3# set interfaces ge-0/0/4 vlan-tagging
user@R3# set interfaces ge-0/0/4 unit 0 vlan-id 10
user@R3# set interfaces ge-0/0/4 unit 0 family inet address 192.168.5.1/24
user@R3# set interfaces ge-0/0/4 unit 0 family iso
user@R3# set interfaces ge-0/0/4 unit 1 vlan-id 11
user@R3# set interfaces ge-0/0/4 unit 1 family inet address 192.168.9.1/24
user@R3# set interfaces ge-0/0/4 unit 1 family iso
user@R3# set interfaces ge-0/0/4 unit 2 vlan-id 12
user@R3# set interfaces ge-0/0/4 unit 2 family inet address 192.168.10.1/24
user@R3# set interfaces ge-0/0/4 unit 2 family iso
```

3. Create the loopback interface and configure the IP and NET addresses.

```
user@R3# set interfaces lo0 unit 0 family inet address 192.168.100.3/32
user@R3# set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0003.00
```

4. Configure policies to redistribute loopback addresses of IS-IS metro-instance (IS-IS metro-a and IS-IS metro-b) and IS-IS standard-instance (core network) routers, so that the routes can be distributed across IS-IS domains as required.

   a. Configure policies to distribute the loopback address of R3.

```
user@R3# set policy-options policy-statement export-direct-loopback from protocol direct
user@R3# set policy-options policy-statement export-direct-loopback from route-filter
192.168.100.3/32 exact
user@R3# set policy-options policy-statement export-direct-loopback then accept
```

   b. Configure policies to distribute the loopback addresses of the R5 and R6 routers (standard IS-IS instance).

```
user@R3# set policy-options policy-statement export-isis from protocol isis
user@R3# set policy-options policy-statement export-isis from level 2
user@R3# set policy-options policy-statement export-isis from route-filter
192.168.100.0/24 longer
user@R3# set policy-options policy-statement export-isis then accept
```

   c. Configure policies to distribute the loopback addresses of R1 (IS-IS metro-a instance).

```
user@R3# set policy-options policy-statement export-isis-metro-a from igp-instance metro-
a
user@R3# set policy-options policy-statement export-isis-metro-a from protocol isis
user@R3# set policy-options policy-statement export-isis-metro-a from level 2
user@R3# set policy-options policy-statement export-isis-metro-a from route-filter
192.168.100.0/24 longer
user@R3# set policy-options policy-statement export-isis-metro-a then accept
```

   d. Configure policies to distribute the loopback addresses of R2 (IS-IS metro-b instance).

```
user@R3# set policy-options policy-statement export-isis-metro-b from igp-instance metro-
b
user@R3# set policy-options policy-statement export-isis-metro-b from protocol isis
```

```
user@R3# set policy-options policy-statement export-isis-metro-b from level 2
user@R3# set policy-options policy-statement export-isis-metro-b from route-filter
192.168.100.0/24 longer
user@R3# set policy-options policy-statement export-isis-metro-b then accept
```

5. Enable IS-IS on the standard-instance interface (connecting R3 to R5) and on the subinterface (connecting R3 to R4).

```
user@R3# set protocols isis interface ge-0/0/3.0 level 1 disable
user@R3# set protocols isis interface ge-0/0/3.0 level 2 metric 100
user@R3# set protocols isis interface ge-0/0/3.0 point-to-point
user@R3# set protocols isis interface ge-0/0/4.0 level 1 disable
user@R3# set protocols isis interface ge-0/0/4.0 level 2 metric 100
user@R3# set protocols isis interface ge-0/0/4.0 point-to-point
user@R3# set protocols isis interface lo0.0 passive
```

6. Configure IS-IS to export loopback addresses from IS-IS metro-a and IS-IS metro-b instances to the IS-IS standard instance. This configuration distributes specific routes instead of flooding the entire metro domain.

```
user@R3# set protocols isis export export-isis-metro-a
user@R3# set protocols isis export export-isis-metro-b
```

7. Enable IS-IS on the IS-IS metro-b instance interface (connecting R3 to R2) and on the subinterface (R3 to R4).

```
user@R3# set protocols isis-instance metro-b interface ge-0/0/1.0 level 1 disable
user@R3# set protocols isis-instance metro-b interface ge-0/0/1.0 level 2 metric 100
user@R3# set protocols isis-instance metro-b interface ge-0/0/1.0 point-to-point
user@R3# set protocols isis-instance metro-b interface ge-0/0/4.2 level 1 disable
user@R3# set protocols isis-instance metro-b interface ge-0/0/4.2 level 2 metric 100
user@R3# set protocols isis-instance metro-b interface ge-0/0/4.2 point-to-point
```

8. Configure IS-IS to export the loopback addresses of IS-IS metro-a and standard IS-IS instances to the IS-IS metro-b instance. This configuration distributes specific routes instead of flooding the entire standard IS-IS instances and metro-a domain instances.

```
user@R3# set protocols isis-instance metro-b export export-isis
user@R3# set protocols isis-instance metro-b export export-direct-loopback
user@R3# set protocols isis-instance metro-b export export-isis-metro-a
```

9. Enable IS-IS on the IS-IS metro-a instance interface (connecting R3 to R1) and on the subinterface (R3 to R4).

```
user@R3# set protocols isis-instance metro-a interface ge-0/0/2.0 level 1 disable
user@R3# set protocols isis-instance metro-a interface ge-0/0/2.0 level 2 metric 100
user@R3# set protocols isis-instance metro-a interface ge-0/0/2.0 point-to-point
user@R3# set protocols isis-instance metro-a interface ge-0/0/4.1 level 1 disable
user@R3# set protocols isis-instance metro-a interface ge-0/0/4.1 level 2 metric 100
user@R3# set protocols isis-instance metro-a interface ge-0/0/4.1 point-to-point
```

10. Configure IS-IS to export the loopback addresses of IS-IS metro-b and standard IS-IS instances to the IS-IS metro-a instance. This configuration distributes specific routes instead of flooding the entire standard IS-IS instances and metro-b domain instances.

```
user@R3# set protocols isis-instance metro-a export export-isis
user@R3# set protocols isis-instance metro-a export export-direct-loopback
user@R3# set protocols isis-instance metro-a export export-isis-metro-b
```

11. Configure routing options to identify the router in the domain.

```
user@R3# set routing-options router-id 192.168.100.3
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, and `show protocols` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
interfaces {
    ge-0/0/1 {
        description R3-to-R2;
        unit 0 {
            family inet {
                address 192.168.3.2/24;
            }
            family iso;
        }
    }
    ge-0/0/2 {
        description R3-to-R1;
        unit 0 {
            family inet {
                address 192.168.2.2/24;
            }
            family iso;
        }
    }
    ge-0/0/3 {
        description R3-to-R5;
        unit 0 {
            family inet {
                address 192.168.6.1/24;
            }
            family iso;
        }
    }
    ge-0/0/4 {
        description R3-to-R4;
        vlan-tagging;
        unit 0 {
            vlan-id 10;
            family inet {
                address 192.168.5.1/24;
            }
            family iso;
```

```
        }
        unit 1 {
            vlan-id 11;
            family inet {
                address 192.168.9.1/24;
            }
            family iso;
        }
        unit 2 {
            vlan-id 12;
            family inet {
                address 192.168.10.1/24;
            }
            family iso;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.100.3/32;
            }
            family iso {
                address 49.0002.0192.0168.0003.00;
            }
        }
    }
}
policy-options {
    policy-statement export-direct-loopback {
        from {
            protocol direct;
            route-filter 192.168.100.3/32 exact;
        }
        then accept;
    }
    policy-statement export-isis {
        from {
            protocol isis;
            level 2;
            route-filter 192.168.100.0/24 longer;
        }
        then accept;
    }
```

```
    policy-statement export-isis-metro-a {
        from {
            igp-instance metro-a;
            protocol isis;
            level 2;
            route-filter 192.168.100.0/24 longer;
        }
        then accept;
    }
    policy-statement export-isis-metro-b {
        from {
            igp-instance metro-b;
            protocol isis;
            level 2;
            route-filter 192.168.100.0/24 longer;
        }
        then accept;
    }
}
protocols {
    isis {
        interface ge-0/0/3.0 {
            level 1 disable;
            level 2 metric 100;
            point-to-point;
        }
        interface ge-0/0/4.0 {
            level 1 disable;
            level 2 metric 100;
            point-to-point;
        }
        interface lo0.0 {
            passive;
        }
        export [ export-isis-metro-a export-isis-metro-b ];
    }
    isis-instance metro-b {
        interface ge-0/0/1.0 {
            level 1 disable;
            level 2 metric 100;
            point-to-point;
        }
        interface ge-0/0/4.2 {
```

```
            level 1 disable;
            level 2 metric 100;
            point-to-point;
        }
        export [ export-isis export-direct-loopback export-isis-metro-a ];
    }
    isis-instance metro-a {
        interface ge-0/0/2.0 {
            level 1 disable;
            level 2 metric 100;
            point-to-point;
        }
        interface ge-0/0/4.1 {
            level 1 disable;
            level 2 metric 100;
            point-to-point;
        }
        export [ export-isis export-direct-loopback export-isis-metro-b ];
    }
}
routing-options {
    router-id 192.168.100.3;
}
```

Verification

IN THIS SECTION

To verify that the configuration is working properly, perform the following tasks:

Verify IS-IS Advertisements

Purpose

Verify the IS-IS advertisement entries in the IS-IS link-state database (LSDB), which contains data about PDU packets.

Action

From operational mode, run the `show isis database level 2` command.

On R3

```
user@R3>show isis database level 2

IS-IS level 2 link-state database:
LSP ID                  Sequence Checksum Lifetime Attributes
R6.00-00                   0x75d   0x1ff7     1181 L1 L2
R5.00-00                   0x75b   0xffdc      741 L1 L2
R4.00-00                   0x780    0x4e1      552 L1 L2
R3.00-00                   0x7f0   0x8643      496 L1 L2
  4 LSPs
```

```
user@R3>show isis database level 2 igp-instance metro-a
IS-IS level 2 link-state database:
LSP ID                  Sequence Checksum Lifetime Attributes
R1.00-00                   0x136   0x46e5     1046 L1 L2
R4.00-00                   0x781   0xf65e      768 L1 L2
R3.00-00                   0x7f2   0x871b      764 L1 L2
  3 LSPs
```

```
user@R3>show isis database level 2 igp-instance metro-b
IS-IS level 2 link-state database:
LSP ID                  Sequence Checksum Lifetime Attributes
R2.00-00                   0x13a   0x7997     1013 L1 L2
R4.00-00                   0x781   0x86ba      771 L1 L2
R3.00-00                   0x7f2   0x1288      510 L1 L2
  3 LSPs
```

On R1

```
user@R1>show isis database level 2

IS-IS level 2 link-state database:
LSP ID                  Sequence Checksum Lifetime Attributes
R1.00-00                   0x136   0x46e5      851 L1 L2
R4.00-00                   0x781   0xf65e      571 L1 L2
R3.00-00                   0x7f2   0x871b      565 L1 L2
  3 LSPs
```

Meaning

This output on R3 illustrates that R3 sees the IS-IS advertisements from R4, R5, and R6 which is standard IS-IS instance. R3 also sees the IS-IS advertisements from R1 (IS-IS metro-a), R2 (IS-IS metro-b), and R4 (both IS-IS metro-a and IS-IS metro-b). Thus, you can see that R3 is a common router that redistributes IS-IS routes among the IS-IS metro-a instance, the IS-IS metro-b instance, and the standard IS-IS instance (core network).

The output on R1 illustrates that R1 sees the IS-IS advertisements only from R3 and R4. R1 does not see any IS-IS advertisements from R2. Thus, you see that IS-IS metro-a and IS-IS metro-b are separate IS-IS flooding domains. You can use this property to build more scalable networks.

Verify the Routing Table

Purpose

Verify the route entries in the routing table.

Action

From operational mode, run the `show route table inet.0 route-destination address extensive` command.

On R3

```
user@R3>show route table inet.0 192.168.100.1 extensive

inet.0: 28 destinations, 28 routes (28 active, 0 holddown, 0 hidden)
192.168.100.1/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 192.168.100.1/32 -> {192.168.2.1}
```

```
IS-IS level 1, LSP fragment 0
IS-IS level 2, LSP fragment 0
IS-IS level 1, LSP fragment 0
IS-IS level 2, LSP fragment 0
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Router, Next hop index: 601
                Address: 0xc5b21cc
                Next-hop reference count: 2
                Next hop: 192.168.2.1 via ge-0/0/2.0, selected
                Session Id: 0x140
                State: <Active Int>
                Age: 2d 18:10:36        Metric: 63
                Validation State: unverified
                ORR Generation-ID: 0
                Task: IS-IS-metro-a
                Announcement bits (3): 0-KRT 2-IS-IS 10-IS-IS-metro-b
                AS path: I
                Thread: junos-main
```

```
user@R3>show route table inet.0 192.168.100.2 extensive
inet.0: 28 destinations, 28 routes (28 active, 0 holddown, 0 hidden)
192.168.100.2/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 192.168.100.2/32 -> {192.168.3.1}
IS-IS level 1, LSP fragment 0
IS-IS level 2, LSP fragment 0
IS-IS level 1, LSP fragment 0
IS-IS level 2, LSP fragment 0
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Router, Next hop index: 602
                Address: 0xc5b2234
                Next-hop reference count: 2
                Next hop: 192.168.3.1 via ge-0/0/1.0, selected
                Session Id: 0x141
                State: <Active Int>
                Age: 2d 18:18:48        Metric: 63
                Validation State: unverified
                ORR Generation-ID: 0
```

```
                Task: IS-IS-metro-b
                Announcement bits (3): 0-KRT 2-IS-IS 4-IS-IS-metro-a
                AS path: I
                Thread: junos-main
```

Meaning

The output illustrates that the loopback address of R1 (192.168.100.1) is mapped to the IS-IS metro-a instance (IS-IS-metro-a) and the loopback address of R2 (192.168.100.2) is mapped to the IS-IS metro-b instance (IS-IS-metro-b) as configured in R3.

Verify the Routes in the IS-IS Routing Table

Purpose

Verify the routes in the IS-IS routing table.

Action

From operational mode, run the `show isis route` command.

On R3

```
user@R3>show isis route
IS-IS routing table            Current version: L1: 1885 L2: 1956
IPv4/IPv6 Routes
----------------
Prefix             L Version   Metric Type Interface       NH   Via                Backup Score
192.168.7.0/24     2    1956      126 int  ge-0/0/4.0      IPV4 R4
192.168.8.0/24     2    1956      126 int  ge-0/0/3.0      IPV4 R5
192.168.100.4/32   2    1956       63 int  ge-0/0/4.0      IPV4 R4
192.168.100.5/32   2    1956       63 int  ge-0/0/3.0      IPV4 R5
192.168.100.6/32   2    1956      126 int  ge-0/0/3.0      IPV4 R5
                                          ge-0/0/4.0      IPV4 R4
```

```
user@R3>show isis route igp-instance metro-a
 IS-IS routing table            Current version: L1: 1889 L2: 1961
IPv4/IPv6 Routes
```

```
----------------
Prefix             L Version   Metric Type Interface       NH   Via                Backup Score
192.168.1.0/24     2    1961      126 int  ge-0/0/4.1      IPV4 R4
                                          ge-0/0/2.0      IPV4 R1
192.168.100.1/32   2    1961       63 int  ge-0/0/2.0      IPV4 R1
```

```
user@R3>show isis route igp-instance metro-b
 IS-IS routing table             Current version: L1: 1892 L2: 1949
IPv4/IPv6 Routes
----------------
Prefix             L Version   Metric Type Interface       NH   Via                Backup Score
192.168.4.0/24     2    1949      126 int  ge-0/0/4.2      IPV4 R4
                                          ge-0/0/1.0      IPV4 R2
192.168.100.2/32   2    1949       63 int  ge-0/0/1.0      IPV4 R2
```

On R1

```
user@R1>show isis route
IS-IS routing table             Current version: L1: 313 L2: 392
IPv4/IPv6 Routes
----------------
Prefix             L Version   Metric Type Interface       NH   Via                Backup Score
192.168.9.0/24     2     392      126 int  ge-0/0/2.0      IPV4 R3
                                          ge-0/0/1.0      IPV4 R4
192.168.100.2/32   2     392      126 int  ge-0/0/2.0      IPV4 R3
                                          ge-0/0/1.0      IPV4 R4
192.168.100.3/32   2     392       73 int  ge-0/0/2.0      IPV4 R3
192.168.100.4/32   2     392       73 int  ge-0/0/1.0      IPV4 R4
192.168.100.5/32   2     392      126 int  ge-0/0/2.0      IPV4 R3
                                          ge-0/0/1.0      IPV4 R4
192.168.100.6/32   2     392      126 int  ge-0/0/2.0      IPV4 R3
                                          ge-0/0/1.0      IPV4 R4
```

Meaning

The output on R3 shows the loopback addresses and the IS-IS instance mapping information of R1, R2, R4, R5, and R6.

The output on R1 shows the loopback addresses of R2, R3, R4, R5, and R6.

Verify IS-IS Interfaces

Purpose

Verify the status information about IS-IS-enabled interfaces.

Action

From operational mode, run the `show isis interface` command.

On R3

```
user@R3>show isis interface
IS-IS interface database:
Interface             L CirID Level 1 DR        Level 2 DR        L1/L2 Metric
ge-0/0/3.0            2   0x1 Disabled          Point to Point         10/100
ge-0/0/4.0            2   0x1 Disabled          Point to Point         10/100
lo0.0                 3   0x1 Passive           Passive                 0/0
```

```
user@R3>show isis interface igp-instance metro-a
IS-IS interface database:
Interface             L CirID Level 1 DR        Level 2 DR        L1/L2 Metric
ge-0/0/2.0            2   0x1 Disabled          Point to Point         10/100
ge-0/0/4.1            2   0x1 Disabled          Point to Point         10/100
```

```
user@R3>show isis interface igp-instance metro-b
IS-IS interface database:
Interface             L CirID Level 1 DR        Level 2 DR        L1/L2 Metric
ge-0/0/1.0            2   0x1 Disabled          Point to Point         10/100
ge-0/0/4.2            2   0x1 Disabled          Point to Point         10/100
```

On R1

```
user@R1>show isis interface
IS-IS interface database:
```

```
Interface             L CirID Level 1 DR       Level 2 DR       L1/L2 Metric
ge-0/0/1.0            2   0x1 Disabled         Point to Point         10/100
ge-0/0/2.0            2   0x1 Disabled         Point to Point         10/100
lo0.0                 3   0x1 Passive          Passive                 0/0
```

Meaning

The output shows the interfaces mapped to different IS-IS instances.

Example: Configure Multiple Independent Instances of OSPFv2 with Segment Routing

SUMMARY

Use this example to configure multiple IGP instances of OSPFv2 with segment routing.

IN THIS SECTION

NOTE: Our content testing team has validated and updated this example.

TIP:

Table 6: Readability Score and Time Estimates

Reading Time	30 minutes

Configuration Time	20 minutes

Example Prerequisites

Hardware requirements	Three MX Series routers.
Software requirements	Junos OS Release 24.4R1 or later running on all devices.

Before You Begin

Benefits	Configuring multiple independent instances of OSPFv2 with segment routing enhances network flexibility, scalability, and control over traffic engineering, especially in large and complex networks.
Know more	Multiple Independent IGP Instances of OSPFv2

Functional Overview

Technologies used	• Routing Protocols:OSPF • Segment Routing with Multiprotocol Label Switching (SR-MPLS) • VLAN Tagging
Primary verification tasks	• Verify that multiple independent OSPF instances are running. • Verify the OSPF segment routing database for different prefix-SIDs advertised by the multiple instances of OSPF

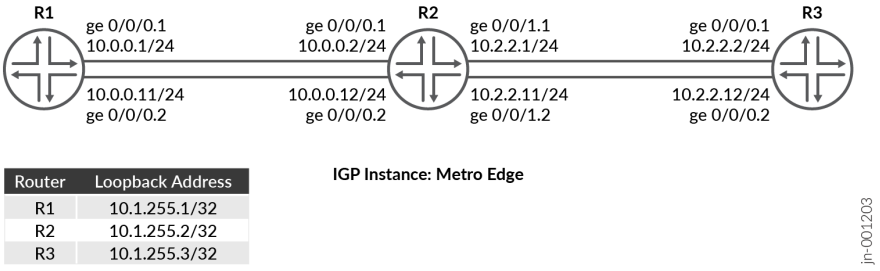
Topology Overview

This configuration example depicts three devices R1, R2, and R3. There are two sub-interfaces configured between device R1 and device R2 and between device R2 and device R3. Each device runs multiple OSPF instances with segment routing enabled. We configure SR-MPLS to provide path control through the network. There are OSPF instances named `metro-edge` running on each of the two subinterfaces of the devices.

Hostname	Role	Function
R1, R2, and R3	The devices have multi-instance OSPF configured on the subinterfaces, with segment routing enabled.	The devices participate in OSPF multi-instances, advertise routes, and forward traffic using prefix-SIDs to other devices.

Topology Illustration

Figure 28:



Router	Loopback Address
R1	10.1.255.1/32
R2	10.1.255.2/32
R3	10.1.255.3/32

R2 Configuration Steps

For complete sample configurations on R2, see: "Appendix 1: Set Commands on All Devices" on page 382

This section highlights the main configuration tasks needed to configure the R2 device for this example.

1. **a.** Configure the basic device settings such as hostname, `enhanced-ip` mode, IPv4 addresses on the logical units of the device interfaces.

   b. Configure the loopback interface with an IP address and enable MPLS.

c. Configure the router ID and autonomous system (AS) number to propagate routing information within a set of routing devices that belong to the same AS.

d. Enable VLAN tagging and configure the logical units of both the interfaces with different VLAN IDs.

e. Enable MPLS on each logical unit. Configure the maximum number of MPLS labels that can be applied to outgoing packets on logical units of each interface.

f. Define a policy to load balance packets and apply the per-packet policy to enable load balancing of traffic.

g. Configure a policy statement that matches routes based on the exact prefix and assign a segment identifier to the matched route.

h. Configure MPLS traffic engineering, segment routing global block (SRGB) label range at the `edit protocol mpls` hierarchy level to ensure the labels are more predictable across segment routing domain, MPLS label range to assign labels from the configured srgb labels for the links.

```
[edit]
set system host-name R2
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 1 family inet address 10.0.0.2/24
set interfaces ge-0/0/0 unit 2 family inet address 10.0.0.12/24
set interfaces ge-0/0/1 unit 1 family inet address 10.2.2.1/24
set interfaces ge-0/0/1 unit 2 family inet address 10.2.2.11/24
```

```
[edit]
set interfaces lo0 unit 0 family inet address 10.1.255.2/32
set interfaces lo0 unit 0 family mpls
```

```
[edit]
set routing-options router-id 10.1.255.2
set routing-options autonomous-system 100
```

```
[edit]
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
set interfaces ge-0/0/0 unit 2 vlan-id 2
set interfaces ge-0/0/1 vlan-tagging
```

```
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 2 vlan-id 2
```

```
[edit]
set interfaces ge-0/0/0 unit 1 family mpls maximum-labels 5
set interfaces ge-0/0/0 unit 2 family mpls maximum-labels 5
set interfaces ge-0/0/1 unit 1 family mpls maximum-labels 5
set interfaces ge-0/0/1 unit 2 family mpls maximum-labels 5
```

```
[edit]
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement pplb then accept
set routing-options forwarding-table export pplb
```

```
[edit]
set policy-options policy-statement prefix-sid term 1 from route-filter 10.1.255.2/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1001
set policy-options policy-statement prefix-sid term 1 then accept
```

```
[edit]
set protocols mpls traffic-engineering
set protocols mpls label-range srgb-label-range 800000 879999
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
```

2. Configure the `ospf-instance metro-edge` on the subinterfaces (connecting from R2 to R1 and from R2 to R3).

```
[edit]
set protocols ospf-instance metro-edge area 0.0.0.0 interface all
```

3. Enable the OSPF `metro-edge` instance to use segment routing with prefix-sids.

```
[edit]
set protocols ospf-instance metro-edge source-packet-routing prefix-segment prefix-sid
```

4. Configure the IPv4 index value of the node segment.

```
[edit]
set protocols ospf-instance metro-edge source-packet-routing node-segment ipv4-index 1
```

5. Configure the loopback address of the OSPF `metro-edge` instance as passive and disable the management interface (fxp0.0).

```
[edit]
set protocols ospf-instance metro-edge area 0.0.0.0 interface lo0.0 passive
set protocols ospf-instance metro-edge area 0.0.0.0 interface fxp0.0 disable
```

Verification

IN THIS SECTION

Command	Verification Task
show route protocol ospf table inet.0 extensive	• Verify the route entries in the routing table. • Verify the loopback address of R1 and R3 is mapped to the igp-instance as configured in R2.

Command	Verification Task
show ospf spring sid-database igp-instance *igp-instance*	Verify the OSPF segment routing database for the OSPF instance.
show ospf neighbor igp-instance *igp-instance*	Verify neighbors for the specific OSPF instance.
show ospf database igp-instance *igp-instance*	Verify the OSPF advertisement entries in the OSPF link-state database (LSDB) associated with the IGP instance.
show ospf interface igp-instance *igp-instance*	Verify the interfaces mapped to the IGP instance.
show ospf route igp-instance *igp-instance*	Verify the routes and OSPF instance mapping information of R1 and R3.

Verify the Routing Table

Purpose

Verify the route entries in the routing table

Action

From operational mode, run the `show route table inet.0 route-destination address extensive` command.

```
user@R2>show route protocol ospf table inet.0 10.1.255.1 extensive
inet.0: 19 destinations, 21 routes (19 active, 0 holddown, 0 hidden)
10.1.255.1/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.1.255.1/32 -> {list:10.0.0.1, 10.0.0.11}
        *OSPF   Preference: 10/10
                Next hop type: Router, Next hop index: 0
                Address: 0x8b32234
                Next-hop reference count: 2, Next-hop session id: 0
                Kernel Table Id: 0
                Next hop: 10.0.0.1 via ge-0/0/0.1, selected
                Session Id: 0
                Next hop: 10.0.0.11 via ge-0/0/0.2
                Session Id: 0
                State: <Active Int>
                Local AS:   100
                Age: 1w4d 16:01:19      Metric: 1
```

```
                Validation State: unverified
                Area: 0.0.0.0
                Task: OSPF-metro-edge
                Announcement bits (1): 0-KRT
                AS path: I
                Thread: junos-main

user@R2>show route protocol ospf table inet.0 10.1.255.3 extensive
inet.0: 19 destinations, 21 routes (19 active, 0 holddown, 0 hidden)
10.1.255.3/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.1.255.3/32 -> {list:10.2.2.2, 10.2.2.12}
        *OSPF   Preference: 10/10
                Next hop type: Router, Next hop index: 0
                Address: 0x8b316f4
                Next-hop reference count: 2, Next-hop session id: 0
                Kernel Table Id: 0
                Next hop: 10.2.2.2 via ge-0/0/1.1, selected
                Session Id: 0
                Next hop: 10.2.2.12 via ge-0/0/1.2
                Session Id: 0
                State: <Active Int>
                Local AS:   100
                Age: 1w4d 16:13:55      Metric: 1
                Validation State: unverified
                Area: 0.0.0.0
                Task: OSPF-metro-edge
                Announcement bits (1): 0-KRT
                AS path: I
                Thread: junos-main
```

Meaning

The output illustrates that the loopback address of R1 (10.1.255.1) and the loopback address of R3 (10.1.255.2) is mapped to the OSPF igp-instance **metro-edge** as configured in R2.

Verify OSPF Advertisements

Purpose

Verify the OSPF advertisement entries in the OSPF link-state database (LSDB) associated with the IGP instance.

Action

From the operational mode, run the `show ospf database igp-instance` *`igp-instance`* command.

```
user@R2>show ospf database igp-instance metro-edge
    OSPF database, Area 0.0.0.0
Type       ID               Adv Rtr           Seq      Age  Opt  Cksum  Len
Router   10.1.255.1       10.1.255.1       0x80000013  1110  0x22 0xe6e9  72
Router  *10.1.255.2       10.1.255.2       0x80000015  1084  0x22 0x7be2  96
Router   10.1.255.3       10.1.255.3       0x80000013  1585  0x22 0x491   72
Network *10.0.0.2         10.1.255.2       0x80000010  2959  0x22 0x6791  32
Network *10.0.0.12        10.1.255.2       0x80000010  2209  0x22 0x3eb   32
Network  10.2.2.2         10.1.255.3       0x80000010  2085  0x22 0x4ba6  32
Network  10.2.2.12        10.1.255.3       0x80000010  1085  0x22 0xe601  32
OpaqArea 7.0.0.1          10.1.255.1       0x80000012   193  0x22 0x8c0   44
OpaqArea*7.0.0.1          10.1.255.2       0x80000012   511  0x22 0x2a9b  44
OpaqArea 7.0.0.1          10.1.255.3       0x80000012   585  0x22 0x4c76  44
OpaqArea 8.0.0.1          10.1.255.1       0x80000010  2610  0x22 0x4683  48
OpaqArea*8.0.0.1          10.1.255.2       0x80000010  2584  0x22 0xac01  52
OpaqArea 8.0.0.1          10.1.255.3       0x80000010  2584  0x22 0x7d06  52
OpaqArea 8.0.0.2          10.1.255.1       0x80000010  1860  0x22 0x4f55  48
OpaqArea*8.0.0.2          10.1.255.2       0x80000011   334  0x22 0xf393  52
OpaqArea 8.0.0.2          10.1.255.3       0x80000011    84  0x22 0xc498  52
OpaqArea*8.0.0.3          10.1.255.2       0x80000010  1834  0x22 0x445a  48
OpaqArea*8.0.0.4          10.1.255.2       0x80000010  1459  0x22 0x4d2c  48
```

Meaning

Verify the Routes in the OSPF Routing Table

Purpose

Verify the routes in the OSPF routing table

Action

From the operational mode, run the `show ospf route` command.

```
user@R2>show ospf route igp-instance metro-edge
Topology default Route Table:

Prefix             Path  Route      NH       Metric NextHop       Nexthop
                   Type  Type       Type             Interface     Address/LSP
10.1.255.1         Intra Router     IP             1 ge-0/0/0.1    10.0.0.1
                                                     ge-0/0/0.2    10.0.0.11
10.1.255.3         Intra Router     IP             1 ge-0/0/1.1    10.2.2.2
                                                     ge-0/0/1.2    10.2.2.12
10.0.0.0/24        Intra Network    IP             1 ge-0/0/0.1
                                                     ge-0/0/0.2
10.1.255.1/32      Intra Network    IP             1 ge-0/0/0.1    10.0.0.1
                                                     ge-0/0/0.2    10.0.0.11
10.1.255.2/32      Intra Network    IP             0 lo0.0
10.1.255.3/32      Intra Network    IP             1 ge-0/0/1.1    10.2.2.2
                                                     ge-0/0/1.2    10.2.2.12
10.2.2.0/24        Intra Network    IP             1 ge-0/0/1.1
                                                     ge-0/0/1.2
299840             Intra Network    Mpls           0 ge-0/0/0.2    10.0.0.11
299840 (S=0)       Intra Network    Mpls           0 ge-0/0/0.2    10.0.0.11
299856             Intra Network    Mpls           0 ge-0/0/0.1    10.0.0.1
299856 (S=0)       Intra Network    Mpls           0 ge-0/0/0.1    10.0.0.1
299904             Intra Network    Mpls           0 ge-0/0/1.2    10.2.2.12
299904 (S=0)       Intra Network    Mpls           0 ge-0/0/1.2    10.2.2.12
299920             Intra Network    Mpls           0 ge-0/0/1.1    10.2.2.2
299920 (S=0)       Intra Network    Mpls           0 ge-0/0/1.1    10.2.2.2
```

Meaning

The output on R2 shows the loopback addresses and OSPF instance mapping information of R1 and R3.

Verify the OSPF segment routing database

Purpose

Verify the OSPF segment routing database for the OSPF instance metro-edge.

Action

From the operational mode, run the `show ospf spring sid-database igp-instance` *`igp-instance`* command.

```
user@R2>show ospf spring sid-database igp-instance metro-edge
    OSPF database, Area 0.0.0.0
SID     Prefix            Advertised-by    Route-type
1000    10.1.255.1/32     10.1.255.1       Intra-Area
1001    10.1.255.2/32     10.1.255.2       Intra-Area
1002    10.1.255.3/32     10.1.255.3       Intra-Area
```

Meaning

The output illustrates the multiple instances of OSPF (metro-edge) advertise prefix-SIDs.

Verify the OSPF Interfaces

Purpose

Verify the status information about OSPF-instance enabled interfaces.

Action

From the operational mode, run the `show ospf interface igp-instance` *`igp-instance`* command.

```
user@R2>show ospf interface igp-instance metro-edge
Interface           State   Area            DR ID           BDR ID          Nbrs
ge-0/0/0.1          DR      0.0.0.0         10.1.255.2      10.1.255.1         1
ge-0/0/0.2          DR      0.0.0.0         10.1.255.2      10.1.255.1         1
ge-0/0/1.1          DR      0.0.0.0         10.1.255.2      10.1.255.3         1
ge-0/0/1.2          DR      0.0.0.0         10.1.255.2      10.1.255.3         1
lo0.0               DRother 0.0.0.0         0.0.0.0         0.0.0.0            0
lo0.0               DRother 0.0.0.0         0.0.0.0         0.0.0.0            0
```

Meaning

The output shows the subinterfaces of R2 mapped to the OSPF instances (metro-edge).

Verify the OSPF Neighbor

Purpose

Verify the adjacencies between the configured links.

Action

From the operational mode, run the `show ospf neighbor igp-instance` *`igp-instance`* command.

```
user@R2>show ospf neighbor igp-instance metro-edge
Address          Interface              State           ID               Pri  Dead
10.0.0.1         ge-0/0/0.1             Full            10.1.255.1       128    35
10.0.0.11        ge-0/0/0.2             Full            10.1.255.1       128    39
10.2.2.2         ge-0/0/1.1             Full            10.1.255.3       128    33
10.2.2.12        ge-0/0/1.2             Full            10.1.255.3       128    36
```

Meaning

Device R2 has established adjacency with Device R1 and Device R3 and as indicated by the State output field which is `Full`.

Appendix 1: Set Commands on All Devices

IN THIS SECTION

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

R1

```
set system host-name R1
set interfaces ge-0/0/0 unit 1 family inet address 10.0.0.1/24
set interfaces ge-0/0/0 unit 2 family inet address 10.0.0.11/24
set interfaces ge-0/0/0 unit 2 enable
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
set interfaces ge-0/0/0 unit 2 vlan-id 2
set interfaces ge-0/0/0 unit 1 family mpls maximum-labels 5
set interfaces ge-0/0/0 unit 2 family mpls maximum-labels 5
set interfaces lo0 unit 0 family inet address 10.1.255.1/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement pplb then accept
set policy-options policy-statement prefix-sid term 1 from route-filter 10.1.255.1/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1000
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 10.1.255.1
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols mpls traffic-engineering
set protocols mpls label-range srgb-label-range 800000 879999
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf-instance metro-edge source-packet-routing prefix-segment prefix-sid
set protocols ospf-instance metro-edge source-packet-routing node-segment ipv4-index 0
set protocols ospf-instance metro-edge area 0.0.0.0 interface all
set protocols ospf-instance metro-edge area 0.0.0.0 interface lo0.0 passive
set protocols ospf-instance metro-edge area 0.0.0.0 interface fxp0.0 disable
```

R2

```
set system host-name R2
set interfaces ge-0/0/0 unit 1 family inet address 10.0.0.2/24
set interfaces ge-0/0/0 unit 2 family inet address 10.0.0.12/24
set interfaces ge-0/0/1 unit 1 family inet address 10.2.2.1/24
set interfaces ge-0/0/1 unit 2 family inet address 10.2.2.11/24
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
```

```
set interfaces ge-0/0/0 unit 2 vlan-id 2
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 2 vlan-id 2
set interfaces ge-0/0/0 unit 1 family mpls maximum-labels 5
set interfaces ge-0/0/0 unit 2 family mpls maximum-labels 5
set interfaces ge-0/0/1 unit 1 family mpls maximum-labels 5
set interfaces ge-0/0/1 unit 2 family mpls maximum-labels 5
set interfaces lo0 unit 0 family inet address 10.1.255.2/32
set interfaces lo0 unit 0 family mpls
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement pplb then accept
set policy-options policy-statement prefix-sid term 1 from route-filter 10.1.255.2/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1001
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 10.1.255.2
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols mpls traffic-engineering
set protocols mpls label-range srgb-label-range 800000 879999
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf-instance metro-edge source-packet-routing prefix-segment prefix-sid
set protocols ospf-instance metro-edge source-packet-routing node-segment ipv4-index 1
set protocols ospf-instance metro-edge area 0.0.0.0 interface all
set protocols ospf-instance metro-edge area 0.0.0.0 interface lo0.0 passive
set protocols ospf-instance metro-edge area 0.0.0.0 interface fxp0.0 disable
```

R3

```
set system host-name R3
set interfaces ge-0/0/0 unit 1 family inet address 10.2.2.2/24
set interfaces ge-0/0/0 unit 2 family inet address 10.2.2.12/24
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 1 vlan-id 1
set interfaces ge-0/0/0 unit 2 vlan-id 2
set interfaces ge-0/0/0 unit 1 family mpls maximum-labels 5
set interfaces ge-0/0/0 unit 2 family mpls maximum-labels 5
set interfaces lo0 unit 0 family inet address 10.1.255.3/32
set interfaces lo0 unit 0 family mpls
```

```
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement pplb then accept
set policy-options policy-statement prefix-sid term 1 from route-filter 10.1.255.3/32 exact
set policy-options policy-statement prefix-sid term 1 then prefix-segment index 1002
set policy-options policy-statement prefix-sid term 1 then accept
set routing-options router-id 10.1.255.3
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols mpls traffic-engineering
set protocols mpls label-range srgb-label-range 800000 879999
set protocols mpls label-range static-label-range 60001 100000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf-instance metro-edge source-packet-routing prefix-segment prefix-sid
set protocols ospf-instance metro-edge source-packet-routing node-segment ipv4-index 2
set protocols ospf-instance metro-edge area 0.0.0.0 interface lo0.0 passive
set protocols ospf-instance metro-edge area 0.0.0.0 interface all
set protocols ospf-instance metro-edge area 0.0.0.0 interface fxp0.0 disable
```

Change History Table

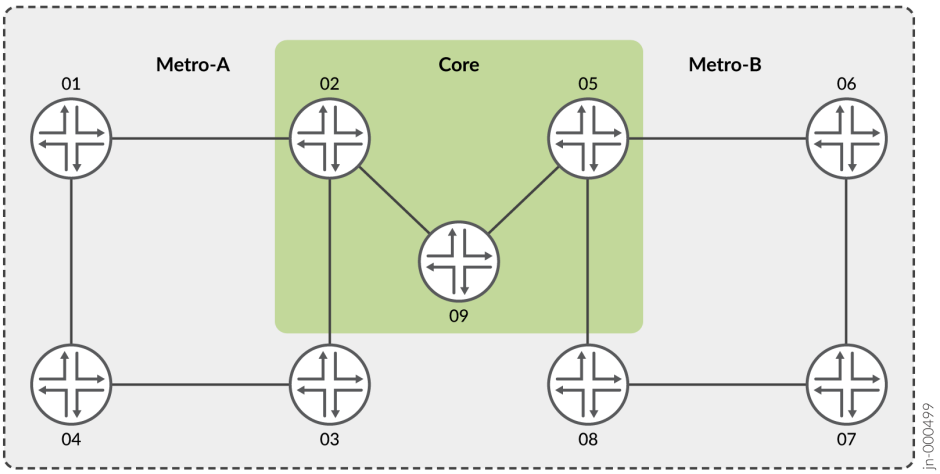
Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
change-completed	Starting in Junos OS and Junos OS Evolved Release 24.2R1, you can configure and run multiple independent IGP instances of OSPFv2 simultaneously on a router.
change-completed	Starting in Junos OS and Junos OS Evolved Release 24.4R1, you can configure and run multiple independent interior gateway protocol (IGP) instances of OSPFv2 with segment routing (SR) on a router.

Flexible Algorithm and Flexible Algorithm Prefix Metrics Leaking across IS-IS Multi-Instance

We've added support to readvertise flexible algorithm (flex algo) prefix-segment identifiers (SIDs) and Flexible Algorithm Prefix Metrics (FAPMs) across interior gateway protocol (IGP) instances. We've also added support to readvertise prefixes from other protocols and assign flex algo prefix-SIDs via policy to those prefixes.

Figure 29: Flexible Algorithm Leaking across IGP Instances



In the sample topology shown in No Link Title, different IS-IS domains, metro-A, metro-B, and the core, constitute a single-segment routing domain. For an end-to-end segment routing flex algo path, nodes 02 and 05 must readvertise flex algo prefix-SIDs and FAPMs across IGP instances.

Flex algo routes are installed in `inet(6)color.0` tables. They could also be installed in colored RIBs, such as `junos-rti-tc-<color>.inet(6).3` when `use-transport-class` statement is configured under `routing-options flex-algorithm <id>`. To support leaking flex algo prefix-SIDs across IGP instances, the `use-transport-class` statement must be configured for that flex algo. Leaking of flex algo prefix-SIDs across IGP instances is policy driven. A sample policy configuration is as follows:

```
[edit policy-options policy-statement name]
user@host# show
from {
    igp-instance <x>; (optional)
    protocol isis; (optional)
    rib <transport-class-rib>;
    route-filter 10.10.10.0/24 orlonger; (optional)
    route-filter 10.20.20.0/24 orlonger; (optional)
    prefix-segment; (optional)
}
then {
    prefix-segment {
        redistribute;
    }
}
```

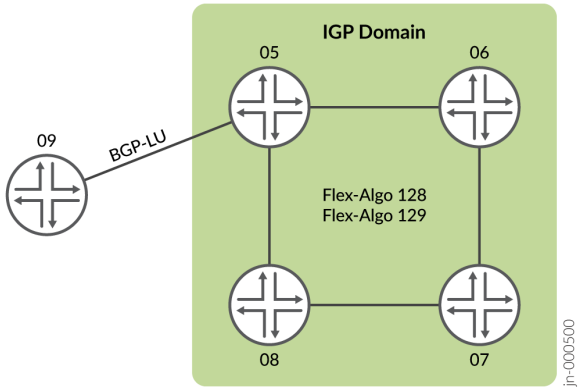
When flex algo prefix-SIDs are leaked across IGP instances, FAPM sub-TLV will be advertised with the metric derived from the export policy or the route's own metric. The metric defined in the export policy has higher precedence over the route's own metric. Additionally, IS-IS installs a stitched route in the mpls.0 tables to stitch incoming MPLS traffic from one IGP instance to the other.

For information on how to apply export policy on multi-instance IS-IS, see export.

Leaking BGP-LU Prefixes into Flexible Algorithm

You can leak BGP-LU prefixes into the IGP with flex algo prefix-SIDs. You can configure the `prefix-segment` (and `metric`) in the `policy-statement` to leak BGP-LU learned prefixes into flex algo.

Figure 30: BGP-LU – Flexible Algorithm Leaking



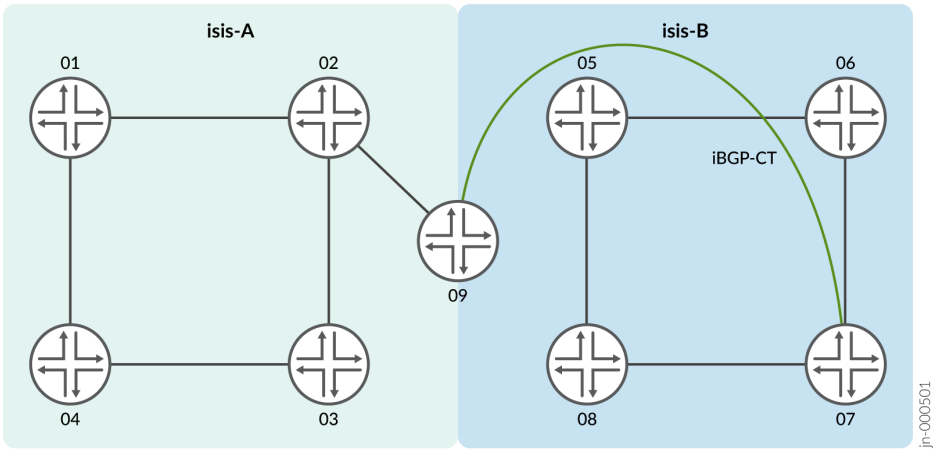
For example, in the topology shown in No Link Title, the IGP domain includes flex algos 128 and 129. The device R9 resides outside the IGP domain. The device R9 is not reachable via flex algo in the IGP domain. Any traffic destined for device R9 follows a flex algo path till device R5 and then follows the device R5 to R9 link.

When flex algo prefix-SIDs are leaked from BGP-LU to an IGP instance, FAPM sub-TLV will be advertised with the metric derived from the export policy or the route's own metric. The metric defined in the export policy has higher precedence over the route's own metric. Additionally, IS-IS installs a stitched route in the mpls.0 tables to stitch incoming MPLS traffic from BGP-LU to IS-IS.

Leaking BGP-CT Prefixes into Flexible Algorithm

You can now leak BGP-CT prefixes into flex algo and vice-versa.

Figure 31: BGP-CT – Flexible Algorithm Leaking



For example, the topology shown in No Link Title consists of multiple IS-IS IGP instances. The ISIS-A has flex algo but does not have BGP-CT. In such deployments, BGP-CT prefixes can be leaked into flex algo and vice-versa via policy configurations.

Currently, BGP-CT prefixes do not support carrying the prefix-SID information. Configure a policy for the prefix and associate a prefix-SID on the router that is redistributing the prefix into IS-IS flex algo.

When flex algo prefix-SIDs are leaked from BGP-CT, FAPM sub-TLV will be advertised with the metric derived from the export policy or the route's metric. The Metric defined in the export policy has higher precedence over the route's metric. Additionally, IS-IS installs a stitched route in the mpls.0 tables to stitch the incoming MPLS traffic from BGP-CT to IS-IS.

Operations and Maintenance

IN THIS SECTION

Operations and Maintenance (SR-MPLS)

Use these Operation, Administration, and Maintenance (OAM) commands to monitor Segment Routing over MPLS (SR-MPLS) traffic and detect connectivity issues.

- ping mpls segment routing isis—Check the operability of MPLS segment routing label-switched path (LSP) connections added by ISIS protocol.
- ping mpls segment routing ospf—Check the operability of MPLS segment routing label-switched path (LSP) connections added by OSPF protocol.
- traceroute mpls segment-routing isis—Trace route to a remote host for a segment routing label-switched path added by the ISIS protocol.
- traceroute mpls segment-routing ospf—Trace route to a remote host for a segment routing label-switched path added by the OSPF protocol.

3

CHAPTER

Segment Routing over IPv6

IN THIS CHAPTER

Overview of SRv6 Network Programming in IS-IS Networks

IN THIS SECTION

Benefits of SRv6 Network Programming

SRv6 network programming provides the following benefits in an IPv6 network:

- Seamless deployment: Network programming depends entirely on the IPv6 header and the header extension to transport a packet, eliminating protocols such as MPLS. This eases deployment without the need for any major hardware or software upgrade in a core IPv6 network.
- Flexible deployment:An SRv6 ingress node can transport packets even if transit routers aren't SRv6-capable. This eliminates the need to deploy segment routing on all nodes in an IPv6 network.
- Single-device versatality: Junos OS supports multiple functions on a single segment identifier (SID) and can inter-operate in the insert mode and the encapsulation mode. This allows a single device to simultaneously perform the provider (P) router and the provider edge (PE) router roles.

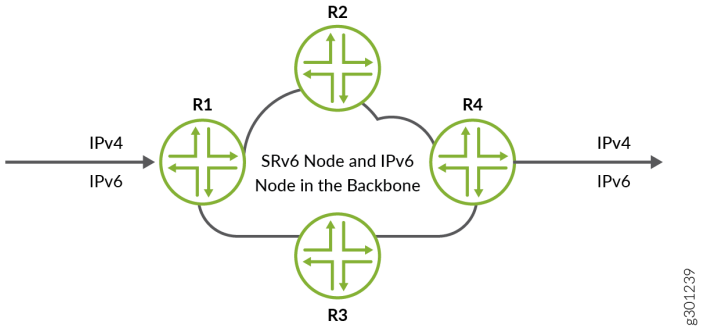
SRv6 Network Programming Overview

Network programming is the capability of a network to encode a network program into individual network instructions that are then inserted into the IPv6 packet headers. The IPv6 packet carrying the network instructions explicitly tells the network about the precise SRv6 nodes available for packet

processing. The network instruction is the SRv6 SID that is represented by a 128-bit IPv6 address. The SIDs are distributed through the network in the IPv6 packet headers. I Along with the addressing, network instructions define a particular task or function for each SRv6-capable node in the SRv6 network.

NOTE: You can configure segment routing in a core IPv6 network without an MPLS data plane on MX Series devices with MPC7E, MPC8E and MPC9E line cards.

This feature is useful for service providers whose networks are predominantly IPv6 and have not deployed MPLS. Such networks depend only on IPv6 headers and header extensions for transmitting data. This feature also benefits networks that need to deploy segment routing traffic through transit routers that do not have segment routing capability yet. In such networks, the SRv6 network programming feature can provide flexibility to leverage segment routing without deploying MPLS.



What Is a Segment Routing Extension Header?

A Segment Identifier represents a specific segment in a segment routing domain. In an IPv6 network, the SID-type used is a 128-bit IPv6 address also referred to as an SRv6 Segment or SRv6 SID. SRv6 stacks up these IPv6 addresses instead of MPLS labels in a segment routing extension header. The Segment Routing Extension Header (SRH) is a type of IPv6 routing extension header. Typically, the SRH contains a segment list encoded as an SRv6 SID. An SRv6 SID consists of the following parts:

- **Locator**—Locator is the first part of an SID that consists of the most significant bits representing the address of a particular SRv6 node. The locator is very similar to a network address that provides a route to its parent node. The IS-IS protocol installs the locator route in the `inet6.0` routing table. IS-IS routes the segment to its parent node, which subsequently performs a function defined in the other part of the SRv6 SID. You can also specify the algorithm associated with this locator. You can define a flexible algorithm as per your network requirements.

- **Function**—The other part of the SID defines a function that is performed locally on the node that is specified by the locator. There are several functions that have already been defined in RFC

8986*Segment Routing over IPv6 (SRv6) Network Programming*. However, the following functions that are signalled in IS-IS are available on Junos OS. . IS-IS installs these function SIDs in the `inet6.0` routing table.

- **End**—An endpoint function for an SRv6 instantiation of a Prefix SID. It does not allow for decapsulation of an outer header for the removal of an SRH. Therefore, an End SID cannot be the last SID of a SID list and cannot be the Destination Address (DA) of a packet without an SRH (unless combined with the PSP, USP or USD flavors).
- **End.X**—An endpoint X function is an SRv6 instantiation of an adjacent SID. It is a variant of the endpoint function with Layer 3 cross-connect to an array of Layer 3 adjacencies.

You can specify an End SID behavior such as Penultimate Segment Pop (PSP), Ultimate Segment Pop (USP), or Ultimate Segment Decapsulation (USD).

- **PSP**—When the last SID is written in the destination address, the End and End.X functions with the PSP flavor pop the top-most SRH. Subsequent stacked SRHs may be present but are not processed as part of the function.
- **USP**—When the next header is an SRH and there are no more segments left, the IS-IS protocol pops the top SRH, looks up the updated destination address and forwards the packet based on match table entry.
- **USD**—When the next header in the packet is 41 or is an SRH and there are no more segments left, then IS-IS pops the outer IPv6 header and its extension headers, looks up the exposed inner IP destination address and forwards the packet to the matched table entry.

NOTE: The size of the locator and function is flexible, and you can customize the size as per your requirements. You must configure the locator before you define the functions. Each locator can advertise multiple end SIDs and end.X SIDs that are associated with it. Ensure that the locator and SIDs belong to the same subnet to avoid commit error.

For example, you can have an SRv6 SID: 2001:DB8:AC05:FF01:A000:: where 2001:DB8:AC05:FF01 represents 64-bit locator and A000 represents 16-bit function:

Table 7: 128-bit SRv6 SID

Locator	Function
2001:db8:AC05:FF01	A000

Flexible Algorithm for SRv6 Dataplane

In a core IPv6 domain configured with segment routing you can define flexible algorithms that compute paths using different parameters and link constraints based on your requirements. For example, you can define a flexible algorithm that computes a path to minimize the IGP metric and define another flexible algorithm to compute a path based on the traffic engineering metrics to divide the network into separate planes. You can configure the flexible algorithm locators to steer packets along the constraint-based paths in an SRv6 domain.

To configure a flexible algorithm for SRv6, see Fleible Algorithms in IS-IS for Segment Routing Traffic Engineering

To advertise the flexible algorithm mapped to the locator, include the `algorithm` option at the `[edit protocols isis source-packet-routing srv6 locator]` hierarchy level. The mapped flexible algorithm is applied to End SIDs and End-X-SIDs under SRv6 locators.

NOTE: If a node is participating in a specific flexible algorithm it applies to both SR MPLS and SRv6 nodes. You cannot define flexible algorithms specifically for either SR MPLS or SRv6.

For ingress traffic, Junos OS uses the encapsulation mode by default. Therefore the destination needs to have USD capable SIDs. Other SRH anchor nodes in the flexible algorithm path can be of any flavor.

For transit traffic in the insert mode, the last anchor node for the flexible algorithm path must have a PSP-capable SID. In the absence of the PSP-capable SID, IS-IS does not download a path through that anchor node. In such cases, IS-IS downloads other ECMP paths with the appropriate flavored SIDs.

TI-LFA for SRv6

Topology Independent- Loop Free Alternate (TI-LFA) establishes a Fast Reroute (FRR) path that is aligned to a post-convergence path. An SRv6-capable node inserts a single segment into the IPv6 header or multiple segments into the SRH. Multiple SRHs can significantly raise the encapsulation overhead, which can sometimes be more than the actual packet payload. Therefore, by default, Junos OS supports SRv6 tunnel encapsulation with reduced SRH. The point-of-local repair (PLR) adds the FRR path information to the SRH containing the SRv6 SIDs.

The TI-LFA backup path is represented as a group of SRv6 SIDs inside an SRH. At the ingress router, IS-IS encapsulates the SRH in a fresh IPv6 header. However, at transit routers, IS-IS inserts the SRH into the data traffic in the following manner:

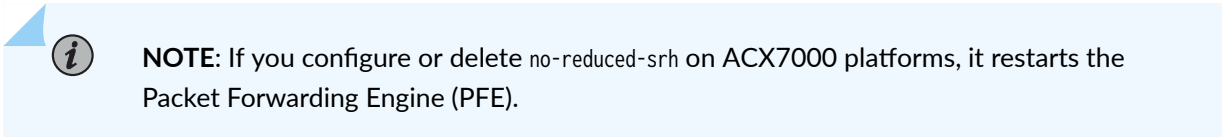
- **Insert Mode**— IS-IS inserts an SRH as the next header in the original IPv6 packet header and modifies the next header according to the value of the SRH. The IPv6 destination address is replaced with the IPv6 address of the first SID in the segment list and the original IPv6 destination address is carried in the SRH header as the last segment in the list. To enable the insert mode at transit routers, include the `transit-srh-insert` statement at the `[edit protocols isis source-packet-routing srv6]` hierarchy level.

- **Encap Mode**— In the encap mode, the original IPv6 packet is encapsulated and transported as the inner packet of an IPv6-in-IPv6 encapsulated packet. The outer IPv6 packet carries the SRH with the segment list. The original IPv6 packet travels unmodified in the network. By default, Junos OS supports SRv6 tunnel encapsulation in reduced SRH. However, you can choose one of the following tunnel encapsulation methods:

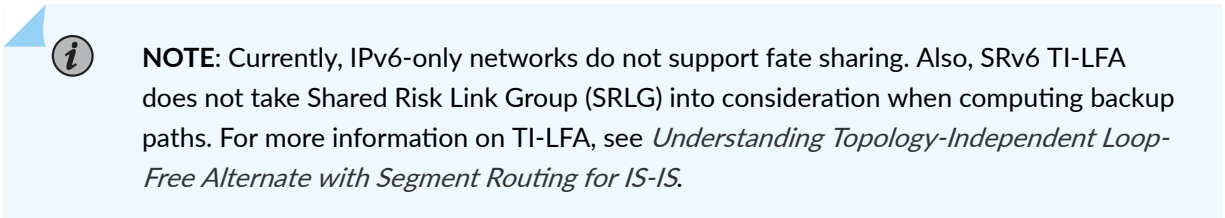
  - **Reduced SRH (default)**— With the reduced SRH mode, if there is only one SID, there is no SRH added and the last SID is copied into the IPV6 destination address. You cannot preserve the entire SID list in the SRH with a reduced SRH.

  - **Non-reduced SRH**— You can configure the non-reduced SRH tunnel encapsulation mode when you want to preserve the entire SID list in the SRH.

    To configure non-reduced SRH, include the `no-reduced-srh` statement at the `[edit routing-options source-packet-routing srv6]` hierarchy level.



NOTE: If you configure or delete `no-reduced-srh` on ACX7000 platforms, it restarts the Packet Forwarding Engine (PFE).



NOTE: Currently, IPv6-only networks do not support fate sharing. Also, SRv6 TI-LFA does not take Shared Risk Link Group (SRLG) into consideration when computing backup paths. For more information on TI-LFA, see *Understanding Topology-Independent Loop-Free Alternate with Segment Routing for IS-IS*.

NSR Support for SRv6

We support IS-IS nonstop active routing (NSR) for dynamic classic adjacency End-x SIDs. Junos OS allocates the same dynamic SID on both the active and backup Routing Engines (RE) after switch-over to ensure dynamically allocated SIDs on the primary RE are not repurposed. You can also use BGP NSR for dynamic DT SIDs. Note that Junos OS currently does not support NSR for classic dynamic End SIDs.

Supported and Unsupported Features for SRv6 Network Programming in IS-IS

SRv6 network programming in IS-IS networks currently supports:

- Core IPv6 and dual stack. IPv4 and IPv6 transport is supported for dual stack.
- IPv4 and IPv6 payloads.

Table 8: Platform-Specific SID Support

Platform	Description	Number of SIDs Supported
ACX7000 family of routers	Maximum number of SIDs supported in SRH	12
	Maximum number of SIDs that can be popped (removed)	7
	Maximum number of SIDs that can be inserted	10
	Maximum number of SIDs supported for encapsulation	12
MX Series	Maximum number of SIDs supported in SRH	6
	Maximum number of SIDs that can be popped (removed)	7
	Maximum number of SIDs that can be inserted	5
	Maximum number of SIDs supported for encapsulation	6
	Maximum number of SIDs supported for End.D (decapsulation) operations	6

Table 8: Platform-Specific SID Support *(Continued)*

Platform	Description	Number of SIDs Supported
PTX10002-36QDD, PTX10008 (Express 5-enhanced mode)	Maximum number of SIDs that can be popped (removed)	7
	Maximum number of SIDs that can be inserted	7
	Maximum number of SIDs supported for encapsulation	7

SRv6 network programming in IS-IS Networks currently does not support:

- Anycast for locator prefix.
- Shared Risk Link Group (SRLG) when computing backup paths.
- Static SRv6 tunnel with segment lists.
- ICMP and ICMPv6 error handling.
- SR-TE policy configuration for SRv6 Tunnel.
- Conflict resolution for Flexible Algorithm locators. Multiple nodes sharing the same locator prefix with different algorithm values could result in unexpected routing behavior.
- Interface group for End-X-SID.
- Configuration of normal or extended admin-groups for IPv6 networks without MPLS. These features can only be configured at [edit protocols mpls] hierarchy level.

RELATED DOCUMENTATION

srv6

locator

flex-algorithm

definition

Example: Configuring SRv6 Network Programming in IS-IS Networks

IN THIS SECTION

This example shows how to configure SRv6 network programing in an IS-IS network. This feature is useful for service providers whose networks are predominantly IPv6 and have not deployed MPLS. Such networks depend only on the IPv6 headers and header extensions for transmitting data. SRv6 network programming provides flexibility to leverage segment routing without deploying MPLS.

Requirements

This example uses the following hardware and software components:

- Eight MX Series routers with MPC7E, MPC8E, or MPC9E line cards
- Junos OS Release 20.3R1 or later

Overview

IN THIS SECTION

You can configure SRv6 without MPLS in a core IPv6 network. This feature benefits networks that need to deploy SR traffic through transit routers that do not have segment routing capability yet.

Topology

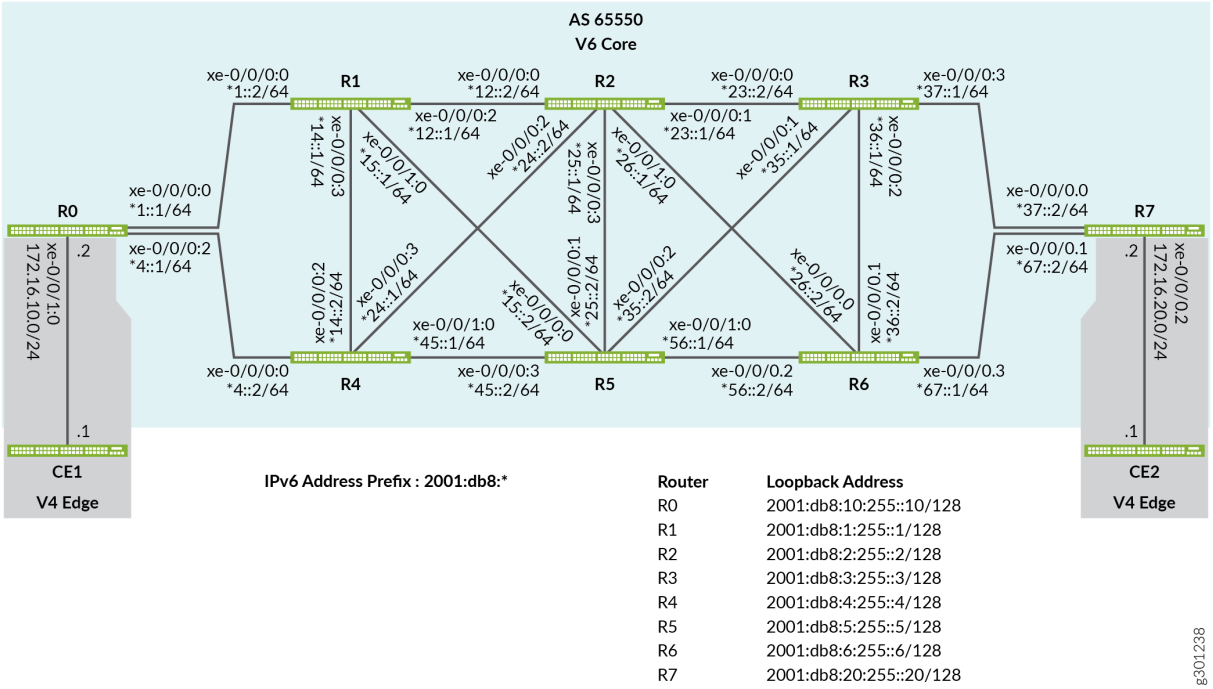
In Figure 32 on page 400, Router R0 and Router R7 are ingress and egress routers that support IPv4 only devices CE1 and CE2. Routers R1, R2, R3, R4, R5, and R6 comprise an IPv6 only provider core network. All routers belong to the same autonomous system. IS-IS is the interior gateway protocol in the IPv6 core and is configured to support SRv6. In this example the Router R2 is configured as an IPv6 route reflector with IBGP peering sessions to both R0 and R7. No other routers speak BGP in this example.

NOTE: To better demonstrate SRv6 tunneling, this example is based on a pure IPv6 provider core. SRv6 is supported with a dual stack core where both IPv6 and IPv4 are deployed.

The edge routers that support IPv4 devices need to transport IPv4 traffic using IPv6 tunnel encapsulation. The encapsulation tunnels are derived from SRv6 SIDs configured at SRv6-enabled routers. The IS-IS protocol processes these SRv6 SIDs and updates the inet6.3 table with the next-hop addresses of the available tunnel endpoints. When an IPv4 route is learned through BGP the router attempts to resolve the associated next hop through the inet6.3 table. When a matching entry is found the result is an automatic IPv6 tunnel to the endpoint that advertised the BGP route.

In this example both the R0 and R7 routers advertise their attached IPv4 subnet using BGP. This results in IPv6 tunnels between the edge routers. The tunnels are used to transport the IPv4 traffic over the IPv6 provider core. At egress, the edge routers decapsulate the outer IPv6 header and perform an IPv4 route lookup to forward the packet to its destination.

Figure 32: SRv6 Network Programming in IS-IS



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter commit from configuration mode.

Router R0

```
set interfaces xe-0/0/0:0 description To_R1
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:1::1/64
set interfaces xe-0/0/0:2 description To_R4
set interfaces xe-0/0/0:2 mtu 4000
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8:4::1/64
set interfaces xe-0/0/1:0 description To_CE1
set interfaces xe-0/0/1:0 unit 0 family inet address 172.16.10.2/24
set interfaces xe-0/0/1:0 unit 0 family iso
set interfaces lo0 unit 0 family iso address 49.0001.000a.0a0a.0a00
set interfaces lo0 unit 0 family inet6 address 2001:db8:10:255::10/128
set policy-options policy-statement pplb then load-balance per-packet
set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a0::/64
set routing-options forwarding-table export pplb
set routing-options router-id 172.16.255.10
set policy-options policy-statement CE1_v4 term 1 from protocol direct
set policy-options policy-statement CE1_v4 term 1 from route-filter 172.16.10.0/24 exact
set policy-options policy-statement CE1_v4 term 1 then next-hop 2001:db8:0:a0:d01::
set policy-options policy-statement CE1_v4 term 1 then accept
set routing-options autonomous-system 65550
set protocols bgp group to-R2RRv6 type internal
set protocols bgp group to-R2RRv6 export CE1_v4
set protocols bgp group to-R2RRv6 local-address 2001:db8:10:255::10
set protocols bgp group to-R2RRv6 neighbor 2001:db8:2:255::2 family inet unicast extended-nexthop
set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a0:1a01:: flavor psp
set protocols isis interface xe-0/0/0:0.0 node-link-protection
set protocols isis interface xe-0/0/0:0.0 point-to-point
set protocols isis interface xe-0/0/0:2.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a0:1a04:: flavor psp
set protocols isis interface xe-0/0/0:2.0 node-link-protection
set protocols isis interface xe-0/0/0:2.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srv6 locator myloc end-sid 2001:db8:0:a0:d01:: flavor
usd
set protocols isis level 1 disable
```

Router R1

```
set interfaces xe-0/0/0:0 description To_R0
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:1::2/64
set interfaces xe-0/0/0:2 description To_R2
set interfaces xe-0/0/0:2 mtu 4000
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8:12:1/64
set interfaces xe-0/0/0:3 description to-R4
set interfaces xe-0/0/0:3 mtu 4000
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8:14::1/64
set interfaces xe-0/0/1:0 description to-R5
set interfaces xe-0/0/1:0 mtu 4000
set interfaces xe-0/0/1:0 unit 0 family iso
set interfaces xe-0/0/1:0 unit 0 family inet6 address 2001:db8:15::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0001.0101.0100
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::1/128
set policy-options policy-statement pplb then load-balance per-packet
set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a1::/64
set routing-options forwarding-table export pplb
set routing-options router-id 192.168.255.1
set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a1:1a10:: flavor psp
set protocols isis interface xe-0/0/0:0.0 node-link-protection
set protocols isis interface xe-0/0/0:0.0 point-to-point
set protocols isis interface xe-0/0/0:2.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a1:1a12:: flavor psp
set protocols isis interface xe-0/0/0:2.0 node-link-protection
set protocols isis interface xe-0/0/0:2.0 point-to-point
set protocols isis interface xe-0/0/0:2.1 node-link-protection
set protocols isis interface xe-0/0/0:3.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a1:1a14:: flavor psp
set protocols isis interface xe-0/0/0:3.0 node-link-protection
set protocols isis interface xe-0/0/0:3.0 point-to-point
set protocols isis interface xe-0/0/1:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a1:1a15:: flavor psp
set protocols isis interface xe-0/0/1:0.0 node-link-protection
set protocols isis interface xe-0/0/1:0.0 point-to-point
set protocols isis interface lo0.0 passive
```

```
set protocols isis source-packet-routing srv6 locator myloc end-sid 2001:db8:0:a1:d11:: flavor
usd
set protocols isis level 1 disable
```

Router R2

```
set interfaces xe-0/0/0:0 description To_R1
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:12::2/64
set interfaces xe-0/0/0:1 description To_R3
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family iso
set interfaces xe-0/0/0:1 unit 0 family inet6 address 2001:db8:23::1/64
set interfaces xe-0/0/0:2 description To_R4
set interfaces xe-0/0/0:2 mtu 4000
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2011:db8:24::1/64
set interfaces xe-0/0/0:3 description To_R5
set interfaces xe-0/0/0:3 mtu 4000
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8:25::1/64
set interfaces xe-0/0/1:0 description To_R6
set interfaces xe-0/0/1:0 mtu 4000
set interfaces xe-0/0/1:0 unit 0 family iso
set interfaces xe-0/0/1:0 unit 0 family inet6 address 2001:db8:26::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0002.0202.0200
set interfaces lo0 unit 0 family inet6 address 2001:db8:2:255::2/128
set policy-options policy-statement pplb then load-balance per-packet
set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a2::/64
set routing-options forwarding-table export pplb
set routing-options router-id 192.168.255.2
set routing-options autonomous-system 65550
set protocols bgp group RRv6 type internal
set protocols bgp group RRv6 local-address 2001:db8:2:255::2
set protocols bgp group RRv6 neighbor 2001:db8:10:255::10 family inet unicast extended-nexthop
set protocols bgp group RRv6 neighbor 2001:db8:20:255::20 family inet unicast extended-nexthop
set protocols bgp group RRv6 cluster 192.168.255.2
set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a2:1a21:: flavor psp
set protocols isis interface xe-0/0/0:0.0 node-link-protection
set protocols isis interface xe-0/0/0:0.0 point-to-point
```

```
set protocols isis interface xe-0/0/0:1.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a2:1a23:: flavor psp
set protocols isis interface xe-0/0/0:1.0 node-link-protection
set protocols isis interface xe-0/0/0:1.0 point-to-point
set protocols isis interface xe-0/0/0:2.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a2:1a24:: flavor psp
set protocols isis interface xe-0/0/0:2.0 node-link-protection
set protocols isis interface xe-0/0/0:2.0 point-to-point
set protocols isis interface xe-0/0/0:3.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a2:1a25:: flavor psp
set protocols isis interface xe-0/0/0:3.0 node-link-protection
set protocols isis interface xe-0/0/0:3.0 point-to-point
set protocols isis interface xe-0/0/1:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a2:1a26:: flavor psp
set protocols isis interface xe-0/0/1:0.0 node-link-protection
set protocols isis interface xe-0/0/1:0.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srv6 locator myloc end-sid 2001:db8:0:a2:d21:: flavor
usd
set protocols isis level 1 disable
```

Router R3

```
set interfaces xe-0/0/0:0 description To_R2
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:23::2/64
set interfaces xe-0/0/0:1 description To_R5
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family iso
set interfaces xe-0/0/0:1 unit 0 family inet6 address 2001:db8:35::1/64
set interfaces xe-0/0/0:2 description To_R6
set interfaces xe-0/0/0:2 mtu 4000
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 36::1/64
set interfaces xe-0/0/0:3 description To_R7
set interfaces xe-0/0/0:3 mtu 4000
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8:37::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0003.0303.0300
set interfaces lo0 unit 0 family inet6 address 2001:db8:3:255::3/128
set policy-options policy-statement pplb then load-balance per-packet
```

```
set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a3::/64
set routing-options forwarding-table export pplb
set routing-options router-id 192.168.255.3
set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a3:1a32:: flavor psp
set protocols isis interface xe-0/0/0:0.0 node-link-protection
set protocols isis interface xe-0/0/0:0.0 point-to-point
set protocols isis interface xe-0/0/0:1.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a3:1a35:: flavor psp
set protocols isis interface xe-0/0/0:1.0 node-link-protection
set protocols isis interface xe-0/0/0:1.0 point-to-point
set protocols isis interface xe-0/0/0:2.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a3:1a36:: flavor psp
set protocols isis interface xe-0/0/0:2.0 node-link-protection
set protocols isis interface xe-0/0/0:2.0 point-to-point
set protocols isis interface xe-0/0/0:3.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a3:1a37:: flavor psp
set protocols isis interface xe-0/0/0:3.0 node-link-protection
set protocols isis interface xe-0/0/0:3.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srv6 locator myloc end-sid 2001:db8:0:a3:d31:: flavor
usd
set protocols isis level 1 disable
```

Router R4

```
set interfaces xe-0/0/0:0 description To_R0
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:4::2/64
set interfaces xe-0/0/0:2 description To_R1
set interfaces xe-0/0/0:2 mtu 4000
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8:14::2/64
set interfaces xe-0/0/0:3 description To_R2
set interfaces xe-0/0/0:3 mtu 4000
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8:24::2/64
set interfaces xe-0/0/1:0 description To_R5
set interfaces xe-0/0/1:0 mtu 4000
set interfaces xe-0/0/1:0 unit 0 family iso
set interfaces xe-0/0/1:0 unit 0 family inet6 address 2001:db8:25::1/64
```

```
set interfaces lo0 unit 0 family iso address 49.0001.0004.0404.0400
set interfaces lo0 unit 0 family inet6 address 2001:db8:4:255::4/128
set policy-options policy-statement pplb then load-balance per-packet
set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a4::/64
set routing-options forwarding-table export pplb
set routing-options router-id 192.168.255.4
set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a4:1a40:: flavor psp
set protocols isis interface xe-0/0/0:0.0 node-link-protection
set protocols isis interface xe-0/0/0:0.0 point-to-point
set protocols isis interface xe-0/0/0:2.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a4:1a41:: flavor psp
set protocols isis interface xe-0/0/0:2.0 node-link-protection
set protocols isis interface xe-0/0/0:2.0 point-to-point
set protocols isis interface xe-0/0/0:3.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a4:1a42:: flavor psp
set protocols isis interface xe-0/0/0:3.0 node-link-protection
set protocols isis interface xe-0/0/0:3.0 point-to-point
set protocols isis interface xe-0/0/1:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a4:1a45:: flavor psp
set protocols isis interface xe-0/0/1:0.0 node-link-protection
set protocols isis interface xe-0/0/1:0.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srv6 locator myloc end-sid 2001:db8:0:a4:d41:: flavor
usd
set protocols isis level 1 disable
```

Router R5

```
set interfaces xe-0/0/0:0 description To_R1
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:15::2/64
set interfaces xe-0/0/0:1 description To_R2
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family iso
set interfaces xe-0/0/0:1 unit 0 family inet6 address 2001:db8:25::2/64
set interfaces xe-0/0/0:2 description To_R3
set interfaces xe-0/0/0:2 mtu 4000
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8:35::2/64
set interfaces xe-0/0/0:3 description To_R4
```

```
set interfaces xe-0/0/0:3 mtu 4000
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8:45::2/64
set interfaces xe-0/0/1:0 description To_R6
set interfaces xe-0/0/1:0 mtu 4000
set interfaces xe-0/0/1:0 unit 0 family iso
set interfaces xe-0/0/1:0 unit 0 family inet6 address 2001:db8:56::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0005.0505.0500
set interfaces lo0 unit 0 family inet6 address 2001:db8:5:255::5/128
set policy-options policy-statement pplb then load-balance per-packet
set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a5::/64
set routing-options forwarding-table export pplb
set routing-options router-id 192.168.255.5
set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a5:1a51:: flavor psp
set protocols isis interface xe-0/0/0:0.0 node-link-protection
set protocols isis interface xe-0/0/0:0.0 point-to-point
set protocols isis interface xe-0/0/0:1.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a5:1a52:: flavor psp
set protocols isis interface xe-0/0/0:1.0 node-link-protection
set protocols isis interface xe-0/0/0:1.0 point-to-point
set protocols isis interface xe-0/0/0:2.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a5:1a53:: flavor psp
set protocols isis interface xe-0/0/0:2.0 node-link-protection
set protocols isis interface xe-0/0/0:2.0 point-to-point
set protocols isis interface xe-0/0/0:3.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a5:1a54:: flavor psp
set protocols isis interface xe-0/0/0:3.0 node-link-protection
set protocols isis interface xe-0/0/0:3.0 point-to-point
set protocols isis interface xe-0/0/1:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a5:1a56:: flavor psp
set protocols isis interface xe-0/0/1:0.0 node-link-protection
set protocols isis interface xe-0/0/1:0.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srv6 locator myloc end-sid 2001:db8:0:a5:d51:: flavor
usd
set protocols isis level 1 disable
```

Router R6

```
set interfaces xe-0/0/0:0 description To_R2
set interfaces xe-0/0/0:0 mtu 4000
```

```
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:26::2/64
set interfaces xe-0/0/0:1 description To_R3
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family iso
set interfaces xe-0/0/0:1 unit 0 family inet6 address 2001:db8:36::2/128
set interfaces xe-0/0/0:2 description To_R5
set interfaces xe-0/0/0:2 mtu 4000
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8:56::2/128
set interfaces xe-0/0/0:3 description To_R7
set interfaces xe-0/0/0:3 mtu 4000
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8:67::1/128
set interfaces lo0 unit 0 family iso address 49.0001.0006.0606.0600
set interfaces lo0 unit 0 family inet6 address 2001:db8:6:255::6/128
set policy-options policy-statement pplb then load-balance per-packet
set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a6::/64
set routing-options forwarding-table export pplb
set routing-options router-id 192.168.255.6
set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a6:1a62:: flavor psp
set protocols isis interface xe-0/0/0:0.0 node-link-protection
set protocols isis interface xe-0/0/0:0.0 point-to-point
set protocols isis interface xe-0/0/0:1.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a6:1a63:: flavor psp
set protocols isis interface xe-0/0/0:1.0 node-link-protection
set protocols isis interface xe-0/0/0:1.0 point-to-point
set protocols isis interface xe-0/0/0:2.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a6:1a65:: flavor psp
set protocols isis interface xe-0/0/0:2.0 node-link-protection
set protocols isis interface xe-0/0/0:2.0 point-to-point
set protocols isis interface xe-0/0/0:3.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a6:1a67:: flavor psp
set protocols isis interface xe-0/0/0:3.0 node-link-protection
set protocols isis interface xe-0/0/0:3.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srv6 locator myloc end-sid 2001:db8:0:a6:d61:: flavor
usd
set protocols isis level 1 disable
```

Router R7

```
set interfaces xe-0/0/0:0 description To_R3
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:37::2/64
set interfaces xe-0/0/0:1 description To_R6
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family iso
set interfaces xe-0/0/0:1 unit 0 family inet6 address 2001:db8:67::2/128
set interfaces xe-0/0/0:2 description To_CE2
set interfaces xe-0/0/0:2 unit 0 family inet address 172.16.20.2/24
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces lo0 unit 0 family iso address 49.0001.0007.0707.0700
set interfaces lo0 unit 0 family inet6 address 2001:db8:20:255::20/32
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement CE2_v4 term 1 from protocol direct
set policy-options policy-statement CE2_v4 term 1 from route-filter 172.16.20.0/24 exact
set policy-options policy-statement CE2_v4 term 1 then next-hop next-hop self
set policy-options policy-statement CE2_v4 term 1 then accept
set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a7::/64
set routing-options forwarding-table export pplb
set routing-options router-id 172.16.255.20
set routing-options autonomous-system 65550
set protocols bgp group to-R2RRv6 type internal
set protocols bgp group to-R2RRv6 local-address 2001:db8:20:255::20
set protocols bgp group to-R2RRv6 neighbor 2001:db8:2:255::2 family inet unicast extended-nexthop
set protocols bgp group to-R2RRv6 export CE2_v4
set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a7:1a73:: flavor psp
set protocols isis interface xe-0/0/0:0.0 node-link-protection
set protocols isis interface xe-0/0/0:0.0 point-to-point
set protocols isis interface xe-0/0/0:1.0 level 2 srv6-adjacency-segment protected locator myloc
end-x-sid 2001:db8:0:a7:1a76:: flavor psp
set protocols isis interface xe-0/0/0:1.0 node-link-protection
set protocols isis interface xe-0/0/0:1.0 point-to-point
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing srv6 locator myloc end-sid 2001:db8:0:a7:d71:: flavor
usd
set protocols isis level 1 disable
```

Configuring Router R0

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see Using the CLI Editor in Configuration Mode in the CLI User Guide.

To configure SRv6 network programming to support IPv4 tunnels over a IPv6 core, perform the following steps on the R0 router:

Step-by-Step Procedure

1. Configure the device interfaces to enable IP transport.

```
[edit]
user@R0# set interfaces xe-0/0/0:0 description To_R1_1
user@R0# set interfaces xe-0/0/0:0 vlan-tagging
user@R0# set interfaces xe-0/0/0:0 unit 0 vlan-id 1
user@R0# set interfaces xe-0/0/0:0 unit 0 family inet address 10.11.1.1/24
user@R0# set interfaces xe-0/0/0:0 unit 0 family iso
user@R0# set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8:1001::1/32
user@R0# set interfaces xe-0/0/0:2 description To_R4_1
user@R0# set interfaces xe-0/0/0:2 vlan-tagging
user@R0# set interfaces xe-0/0/0:2 unit 0 vlan-id 1
user@R0# set interfaces xe-0/0/0:2 unit 0 family inet address 10.21.1.1/24
user@R0# set interfaces xe-0/0/0:2 unit 0 family iso
user@R0# set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8:2021::1/32
user@R0# set interfaces xe-0/0/1:0 description to_RT
user@R0# set interfaces xe-0/0/1:0 vlan-tagging
user@R0# set interfaces xe-0/0/1:0 unit 1 vlan-id 1
user@R0# set interfaces xe-0/0/1:0 unit 1 family inet address 172.20.1.1/24
user@R0# set interfaces xe-0/0/1:0 unit 1 family iso
user@R0# set interfaces xe-0/0/1:0 unit 1 family inet6 address 2001:db8::20:1:1:1/120
user@R0# set interfaces xe-0/0/1:0 unit 4 vlan-id 4
user@R0# set interfaces xe-0/0/1:0 unit 4 family inet address 172.20.2.1/24
user@R0# set interfaces xe-0/0/1:0 unit 4 family iso
user@R0# set interfaces xe-0/0/1:0 unit 4 family inet6 address 2001:db8::20:2:1:1/120
```

2. Configure the loopback interface with IPv4 and IPv6 addresses that is used as router ID for BGP sessions.

```
[edit]
user@R0# set interfaces lo0 unit 0 family inet address 192.168.0.10/32
user@R0# set interfaces lo0 unit 0 family iso address 49.0001.000a.0a0a.0a00
user@R0# set interfaces lo0 unit 0 family inet6 address 2001:db8::10:10:10:10/32
```

3. Configure the router ID and autonomous system (AS) number to propagate routing information within a set of routing devices that belong to the same AS.

```
[edit]
user@R0# set routing-options router-id 10.10.10.10
user@R0# set routing-options autonomous-system 65550
```

4. Enable SRv6 globally and the locator address to indicate the SRv6 capability of the router. SRv6 SID is an IPv6 address that consists of the locator and a function. The routing protocols advertise the locator addresses.

```
[edit]
user@R0# set routing-options source-packet-routing srv6 locator myloc 2001:db8:0:a0::/64
```

5. Configure the End-SID function for the prefix segments. Specify a flavor, that is the behavior of the End-SID function as per your network requirements. Penultimate Segment Pop (PSP), Ultimate Segment Pop (USP), and Ultimate Segment Decapsulation (USP) are the three available flavors for SRv6 functions.

NOTE: Ensure that the locator and the End-SID are in the same subnet to avoid a commit error.

```
[edit]
user@R0# set protocols isis source-packet-routing srv6 locator myloc end-sid
2001:db8:0:a0:d01:: flavor usd
user@R0# set protocols isis source-packet-routing srv6 locator myloc1 end-sid
2001:db8:0:a10:d01:: flavor usd
user@R0# set protocols isis source-packet-routing srv6 locator myloc2 end-sid
2001:db8:0:a20:d01:: flavor usd
user@R0# set protocols isis source-packet-routing srv6 locator myloc3 end-sid
```

```
2001:db8:0:a30:d01:: flavor usd
user@R0# set protocols isis source-packet-routing srv6 locator myloc4 end-sid
2001:db8:0:a40:d01:: flavor usp
user@R0# set protocols isis source-packet-routing srv6 locator myloc4 end-sid
2001:db8:0:a40:d01:: flavor usd
user@R0# set protocols isis level 1 disable
```

6. Configure End-X-SID function on the point-to-point (P2P) interface for the adjacency segments. Specify one or more flavor for the End-X-SID.

NOTE: Ensure that the Locator and End-X-SID are in the same subnet to avoid a commit error. You must enable SRv6 and configure the locator at the `[edit routing-options]` before mapping locators to interfaces.Whenever you configure an `srv6-adjacency-segment` you must also configure the related locator under the `protocols isis source-packet-routing srv6 locator` hierarchy, as shown in step 5.

```
[edit]
user@R0# set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected
locator myloc end-x-sid 2001:db8:0:a0:1a01:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected
locator myloc1 end-x-sid 2001:db8:0:a10:1a01:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected
locator myloc2 end-x-sid 2001:db8:0:a20:1a01:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected
locator myloc3 end-x-sid 2001:db8:0:a30:1a01:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:0.0 level 2 srv6-adjacency-segment protected
locator myloc4 end-x-sid 2001:db8:0:a40:1a01:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:0.0 node-link-protection
user@R0# set protocols isis interface xe-0/0/0:0.0 point-to-point
```

7. Configure SRv6 options for the adjacency segment of the LAN interface xe-0/0/0:2.0. Specify a flavor as per your network requirements. Penultimate Segment Pop (PSP), Ultimate Segment Pop (USP), and Ultimate Segment Decapsulation (USP) are the three available flavors for the SRv6 adjacency segment.

NOTE: Ensure that the Locator and End-X-Sid are in the same subnet to avoid a commit error. You must enable SRv6 and configure the locator at the `[edit routing-options]` before mapping locators to interfaces.

```
[edit]
user@R0# set protocols isis interface xe-0/0/0:2.0 level 2 lan-neighbor 0100.0404.0404 srv6-
adjacency-segment unprotected locator myloc end-x-sid 2001:db8:0:a0:1a04:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:2.0 level 2 lan-neighbor 0100.0404.0404 srv6-
adjacency-segment unprotected locator myloc1 end-x-sid 2001:db8:0:a10:1a04:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:2.0 level 2 lan-neighbor 0100.0404.0404 srv6-
adjacency-segment unprotected locator myloc2 end-x-sid 2001:db8:0:a20:1a04:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:2.0 level 2 lan-neighbor 0100.0404.0404 srv6-
adjacency-segment unprotected locator myloc3 end-x-sid 2001:db8:0:a30:1a04:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:2.0 level 2 lan-neighbor 0100.0404.0404 srv6-
adjacency-segment unprotected locator myloc4 end-x-sid 2001:db8:0:a40:1a04:: flavor usd
user@R0# set protocols isis interface xe-0/0/0:2.0 node-link-protection
user@R0# set protocols isis interface xe-0/0/1:0.1
user@R0# set protocols isis interface fxp0.0 disable
user@R0# set protocols isis interface lo0.0 passive
```

8. Configure BGP on the core-facing interface to establish internal peering sessions.

```
[edit]
user@R0# set protocols bgp group to-PEv6 type internal
user@R0# set protocols bgp group to-PEv6 local-address abcd::10:10:10:10
user@R0# set protocols bgp group to-PEv6 neighbor abcd::2:2:2:2 family inet unicast
extended-nexthop
user@R0# set protocols bgp group to-PE2 type internal
user@R0# set protocols bgp group to-PE2 local-address 10.10.10.10
user@R0# set protocols bgp group to-PE2 neighbor 2.2.2.2 family inet6 unicast
user@R0# set protocols bgp group to-PE2 neighbor 2.2.2.2 family inet6-vpn unicast
```

9. Define a policy to load balance packets.

```
[edit]
user@R0# set policy-options policy-statement pplb then load-balance per-packet
```

10. Apply the per-packet policy to enable load balancing of traffic.

```
[edit]
user@R0# set routing-options forwarding-table export pplb
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@R0# show interfaces
xe-0/0/0:0 {
    description To_R1;
    mtu 4000;
    unit 0 {
        family iso;
        family inet6 {
            address 2001:db8:1::1/64;
        }
    }
}
xe-0/0/0:2 {
    description To_R4;
    mtu 4000;
    unit 0 {
        family iso;
        family inet6 {
            address 2001:db8:4::1/64;
        }
    }
}
xe-0/0/1:0 {
    description To_CE1;
    unit 0 {
        family inet {
            address 172.16.10.2/24;
        }
        family iso;
```

```
    }
}
lo0 {
    unit 0 {
        family iso {
            address 49.0001.000a.0a0a.0a00;
        }
        family inet6 {
            address 2001:db8:10:255::10/128;
        }
    }
}
```

```
[edit]
user@R0# show protocols
bgp {
    group to-R2RRv6 {
        type internal;
        local-address 2001:db8:10:255::10;
        export CE1_v4;
        neighbor 2001:db8:2:255::2 {
            family inet {
                unicast {
                    extended-nexthop;
                }
            }
        }
    }
}
isis {
    interface xe-0/0/0:0.0 {
        level 2 {
            srv6-adjacency-segment {
                protected {
                    locator myloc {
                        end-x-sid 2001:db8:0:a0:1a01:: {
                            flavor psp;
                        }
                    }
                }
            }
```

```
        }
        node-link-protection;
        point-to-point;
    }
    interface xe-0/0/0:2.0 {
        level 2 {
            srv6-adjacency-segment {
                protected {
                    locator myloc {
                        end-x-sid 2001:db8:0:a0:1a04:: {
                            flavor psp;
                        }
                    }
                }
            }
        }
        node-link-protection;
        point-to-point;
    }
    interface lo0.0 {
        passive;
    }
    source-packet-routing {
        srv6 {
            locator myloc {
                end-sid 2001:db8:0:a0:d01:: {
                    flavor {
                        usd;
                    }
                }
            }
        }
    }
    level 1 disable;
}
```

```
[edit]
user@R0# show policy-options
policy-statement CE1_v4 {
    term 1 {
        from {
```

```
            protocol direct;
            route-filter 172.16.10.0/24 exact;
        }
        then {
            next-hop 2001:db8:0:a0:d01::;
            accept;
        }
    }
}
policy-statement pplb {
    then {
        load-balance per-packet;
    }
}
```

```
[edit]
user@R0# show routing-options
source-packet-routing {
    srv6 {
        locator myloc 2001:db8:0:a0::/64;
    }
}
forwarding-table {
    export pplb;
}
router-id 172.16.255.10;
autonomous-system 65550;
```

When done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying IS-IS Adjacency and IBGP Session

Purpose

Verify IS-IS adjacencies and IBGP session at R2. R2 is chosen for this task because it has 5 adjacencies and also serves as the router reflector for the BGP control plane.

NOTE: Its a good idea to confirm the IS-IS adjacencies on all routers before proceeding to the remaining verification steps. A successful SRv6 deployment requires that the interior gateway protocol is operational on all nodes.

Action

From operational mode, run the **show isis adjacency** command on router R2.

```
user@R2> show isis adjacency
Interface             System         L State       Hold (secs) SNPA
xe-0/0/0:0.0          R1             2  Up                   26
xe-0/0/0:1.0          R3             2  Up                   25
xe-0/0/0:2.0          R4             2  Up                   25
xe-0/0/0:3.0          R5             2  Up                   24
xe-0/0/1:0.0          R6             2  Up                   18
```

From operational mode, run the **show bgp summary** command on router R2.

```
user@R2> show bgp summary
Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       2          2          0          0          0          0
inet6.0
                       0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
2001:db8:10:255::10       65550       3101       3092       0       0    23:14:18 Establ
  inet.0: 1/1/1/0
2001:db8:20:255::20       65550       3091       3080       0       0    23:10:10 Establ
  inet.0: 1/1/1/0
```

Meaning

The output confirms the expected IS-IS adjacency count for the R2 router. It also confirms that R2 has established IPv6 based BGP sessions to both the R0 and R7 routers.

Verify SRv6 is Enabled

Purpose

Verify that SRv6 is enabled with a locator, End-SID, and flavor on Router R0.

Action

From operational mode, run the **show isis overview** command on Router R0.

```
user@R0> show isis overview
Instance: master
  Router ID: 172.16.255.10
  IPv6 Router ID: 2001:db8:1::1
  Hostname: R0
  Sysid: 0100.0a0a.0a0a
  Areaid: 49.00
```

```
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
Traffic engineering: enabled
Restart: Disabled
  Helper mode: Enabled
Layer2-map: Disabled
Source Packet Routing (SPRING): Enabled
  Node Segments: Disabled
 SRv6: Enabled
   Locator: 2001:db8:0:a0::/64, Algorithm: 0
      END-SID: 2001:db8:0:a0:d01::, Flavor: USD
Post Convergence Backup: Disabled
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, Narrow metrics are enabled
  Source Packet Routing is enabled
```

Meaning

The configured SRv6 locator `SRv6: Enabled Locator: 2001:db8:0:a0::/64, Algorithm: 0` and , End-SID and flavor `END-SID: 2001:db8:0:a0:d01::, Flavor: USD` are displayed in the output.

Verify the SRv6 End-X-SID Configuration

Purpose

Verify that an End-X-SID function and flavor are configured on R0.

Action

From operational mode, run the **show isis adjacency detail** command on Router R0.

```
user@R0> show isis adjacency detail
R1
  Interface: xe-0/0/0:0.0, Level: 2, State: Up, Expires in 19 secs
  Priority: 0, Up/Down transitions: 1, Last transition: 03:51:48 ago
  Circuit type: 2, Speaks: IP, IPv6
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  IP addresses: 192.168.255.1
  IPv6 addresses: fe80::2e6b:f5ff:fedb:e800
  IPv6 Global Interface Address: 2001:db8:1::2
  Level 2 SRv6 protected END-X-SID: 2001:db8:0:a0:1a01::
    Flavor: PSP, Flags: B-P, Algorithm: 0

R4
  Interface: xe-0/0/0:2.0, Level: 2, State: Up, Expires in 20 secs
  Priority: 0, Up/Down transitions: 1, Last transition: 03:48:04 ago
  Circuit type: 2, Speaks: IP, IPv6
  Topologies: Unicast
  Restart capable: Yes, Adjacency advertisement: Advertise
  IP addresses: 192.168.255.4
  IPv6 addresses: fe80::2e6b:f5ff:feb4:4000
  IPv6 Global Interface Address: 2001:db8:4::2
  Level 2 SRv6 protected END-X-SID: 2001:db8:0:a0:1a04::
    Flavor: PSP, Flags: B-P, Algorithm: 0
```

Meaning

The field `SRv6 protected END-X-SID: 2001:db8:0:a0:1a01::` indicates that End-X-SID function with `Flavor PSP` has been configured on router R0 for the interface used to attach to R1. Similar output is confirmed for the interface connected to R4, which uses a different End-X-SID.

Verifying the Locator Route is Installed

Purpose

Verify that the locator route has been installed.

Action

From operational mode, run the **show route 2001:db8:0:a0::/64 detail** command on router R0.

```
user@R0> show route 2001:db8:0:a0::/64 detail
inet6.0: 75 destinations, 75 routes (75 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8:0:a0::/64*[IS-IS/18] 3d 19:03:16, metric 0
                      Reject

user@R0> show route 2001:db8:0:a0::/64 detail
inet6.0: 45 destinations, 45 routes (45 active, 0 holddown, 0 hidden)
2001:db8:0:a0::/64 (1 entry, 1 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Reject, Next hop index: 0
                Address: 0xc54526c
                Next-hop reference count: 2
                State: <Active Int OpaqueData>
                Local AS: 65550
                Age: 22:15:32   Metric: 0
                Validation State: unverified
                ORR Generation-ID: 0
                Task: IS-IS
                Announcement bits (2): 0-KRT 5-Resolve tree 5
                AS path: I
        . . .
```

Meaning

The output confirms the locator route `2001:db8:0:a0::/64*[IS-IS/18]` is installed in the `inet6.0` table.

Verifying the End-X-SID Route is Installed

Purpose

To display the configured End-X-SID route information that is applied at the interface.

Action

From operational mode, run the **show route 2001:db8:0:a0:1a01::** command on Router R0.

```
user@R0> show route 2001:db8:0:a0:1a01::
inet6.0: 45 destinations, 45 routes (45 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8:0:a0::1a01/128
                   *[IS-IS/18] 04:33:42, metric 0
                    >  to fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0
```

Meaning

The output confirms the End-X-SID route `2001:db8:0:a0::1a01/128` is installed in the `inet.6.0` routing table.

Verifying the End-SID Route is Installed

Purpose

Verify that the End-SID routes for all routers in the SRv6 domain are installed in the `inet6.3` table at Router R0.

Action

From operational mode, run the **show route table inet6.3 protocol isis** command on Router R0 to see all End-SIDs the router has learned. Then display detailed information about the End-SID associated with the R7 router.

```
user@R0> show route table inet6.3 protocol isis
inet6.3: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8:0:a1::d11/128
                   *[SRV6-ISIS/14] 04:39:22, metric 10
                    >  to fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0, SRV6-Tunnel, Dest:
2001:db8:0:a1:d11::
2001:db8:0:a2::d21/128
                   *[SRV6-ISIS/14] 04:35:38, metric 20
                       to fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0, SRV6-Tunnel, Dest:
```

```
2001:db8:0:a2:d21::
                    >  to fe80::2e6b:f5ff:feb4:4000 via xe-0/0/0:2.0, SRV6-Tunnel, Dest:
2001:db8:0:a2:d21::
2001:db8:0:a3::d31/128
                   *[SRV6-ISIS/14] 04:35:38, metric 30
                       to fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0, SRV6-Tunnel, Dest:
2001:db8:0:a3:d31::
                    >  to fe80::2e6b:f5ff:feb4:4000 via xe-0/0/0:2.0, SRV6-Tunnel, Dest:
2001:db8:0:a3:d31::
2001:db8:0:a4::d41/128
                   *[SRV6-ISIS/14] 04:35:38, metric 10
                    >  to fe80::2e6b:f5ff:feb4:4000 via xe-0/0/0:2.0, SRV6-Tunnel, Dest:
2001:db8:0:a4:d41::
2001:db8:0:a5::d51/128
                   *[SRV6-ISIS/14] 04:35:01, metric 20
                       to fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0, SRV6-Tunnel, Dest:
2001:db8:0:a5:d51::
                    >  to fe80::2e6b:f5ff:feb4:4000 via xe-0/0/0:2.0, SRV6-Tunnel, Dest:
2001:db8:0:a5:d51::
2001:db8:0:a6::d61/128
                   *[SRV6-ISIS/14] 04:34:32, metric 30
                       to fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0, SRV6-Tunnel, Dest:
2001:db8:0:a6:d61::
                    >  to fe80::2e6b:f5ff:feb4:4000 via xe-0/0/0:2.0, SRV6-Tunnel, Dest:
2001:db8:0:a6:d61::
2001:db8:0:a7::d71/128
                   *[SRV6-ISIS/14] 04:33:00, metric 40
                       to fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0, SRV6-Tunnel, Dest:
2001:db8:0:a7:d71::
                    >  to fe80::2e6b:f5ff:feb4:4000 via xe-0/0/0:2.0, SRV6-Tunnel, Dest:
2001:db8:0:a7:d71::

user@R0> show route 2001:db8:0:a7::d71/128 detail

inet6.3: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
2001:db8:0:a7::d71/128 (1 entry, 1 announced)
        *SRV6-ISIS Preference: 14
                Level: 2
                Next hop type: List, Next hop index: 1048577
                Address: 0xdb8deb4
                Next-hop reference count: 6
                Next hop: ELNH Address 0xc5462d4 weight 0x1
                    Next hop type: Chain, Next hop index: 582
```

```
                    Address: 0xc5462d4
                    Next-hop reference count: 1
                    Next hop: ELNH Address 0xc545bcc
                    SRV6-Tunnel: Reduced-SRH Encap-mode
                     Src: 2001:db8:1::1 Dest: 2001:db8:0:a7:d71::
                     Segment-list[0] 2001:db8:0:a7:d71::
                        Next hop type: Router, Next hop index: 580
                        Address: 0xc545bcc
                        Next-hop reference count: 9
                        Next hop: fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0 weight 0x1
                Next hop: ELNH Address 0xc546338 weight 0x1, selected
                    Next hop type: Chain, Next hop index: 583
                    Address: 0xc546338
                    Next-hop reference count: 1
                    Next hop: ELNH Address 0xc545f50
                    SRV6-Tunnel: Reduced-SRH Encap-mode
                     Src: 2001:db8:1::1 Dest: 2001:db8:0:a7:d71::
                     Segment-list[0] 2001:db8:0:a7:d71::
                        Next hop type: Router, Next hop index: 581
                        Address: 0xc545f50
                        Next-hop reference count: 9
                        Next hop: fe80::2e6b:f5ff:feb4:4000 via xe-0/0/0:2.0 weight 0x1
                State: <Active NoReadvrt Int OpaqueData>
                Local AS: 65550
                Age: 4:35:43    Metric: 40
                Validation State: unverified
                ORR Generation-ID: 0
                Task: IS-IS
                Announcement bits (3): 0-Resolve tree 2 1-Resolve tree 5 2-Resolve_IGP_FRR task
                AS path: I
                Session-IDs associated:
                Session-id: 322 Version: 0
```

Meaning

The output confirms that Router R0 has learned End-SIDs, that is, `2001:db8:0:a1::d11/128` and `2001:db8:0:a2::d21/128`, from all other routers in the topology. Note the End-SIDs have been installed in the `inet6.3` table. The detailed output for the End-SID advertised by R7 `2001:db8:0:a7:d71::` confirms an SRv6 tunnel has been established between Router R0 and Router R7.

Note that the segment list is populated with the End-SID value configured on the Router R7. Recall that all End-SIDs in this example are configured with the Ultimate Segment Decapsulate (USD) flavor. It's the

combination of a local End-SID and the associated USD flavor that tells R7 it's the egress of the IPv6 tunnel. Upon receipt R7 decapsulates the IPv4 packet and routes it according to the IPv4 destination address.

Verify the SRv6 Configuration in the IS-IS Database

Purpose

Display the IS-IS database to verify the End-SID and flavor configured at Router R7. In this example the command is executed on Router R0. Similar output is expected on all router because the IS-IS database is replicated to all nodes.

Action

From operational mode, run the **show isis database R7.00-00 extensive** command on Router R0.

```
user@R0> show isis database R.00-00 extensive
IS-IS level 1 link-state database:

IS-IS level 2 link-state database:

R7.00-00 Sequence: 0x31f, Checksum: 0x2ce6, Lifetime: 904 secs
   IS neighbor: R3.00                          Metric:       10
     Two-way fragment: R3.00-00, Two-way first fragment: R3.00-00
   IS neighbor: R6.00                          Metric:       10
     Two-way fragment: R6.00-00, Two-way first fragment: R6.00-00
   V6 prefix: 2001:db8::/32                    Metric:        0 Internal Up
   V6 prefix: 2001:db8:0:a7::/64               Metric:        0 Internal Up
   V6 prefix: 2001:db8:20:255::20/128          Metric:        0 Internal Up
   V6 prefix: 2001:db8:37::/64                 Metric:       10 Internal Up
   V6 prefix: 2001:db8:67::2/128               Metric:       10 Internal Up

  Header: LSP ID: R7.00-00, Length: 445 bytes
    Allocated length: 746 bytes, Router ID: 172.16.255.20
    Remaining lifetime: 904 secs, Level: 2, Interface: 360
    Estimated free bytes: 0, Actual free bytes: 301
    Aging timer expires in: 904 secs
    Protocols: IP, IPv6

  Packet: LSP ID: R7.00-00, Length: 445 bytes, Lifetime : 1192 secs
    Checksum: 0x2ce6, Sequence: 0x31f, Attributes: 0x3 <L1 L2>
```

```
    NLPID: 0x83, Fixed length: 27 bytes, Version: 1, Sysid length: 0 bytes
    Packet type: 20, Packet version: 1, Max area: 0

  TLVs:
    Area address: 49.00 (2)
    LSP Buffer Size: 1492
    Speaks: IP
    Speaks: IPV6
    IP router id: 172.16.255.20
    IP address: 172.16.255.20
    IPv6 TE Router ID: 2001:db8:20:255::20
    Hostname: R7
    SRv6 Locator: 2001:db8:0:a7::/64, Metric: 0, MTID: 0, Flags: 0x0, Algorithm: 0
      SRv6 SID: 2001:db8:0:a7:d71::, Flavor: USD
    IPv6 prefix: 2001:db8:20:255::20/128 Metric 0 Up
    IPv6 prefix: 2001:db8::/32 Metric 0 Up
    IPv6 prefix: 2001:db8:0:a7::/64 Metric 0 Up
    IPv6 prefix: 2001:db8:37::/64 Metric 10 Up
    IPv6 prefix: 2001:db8:67::2/128 Metric 10 Up
    Router Capability:  Router ID 172.16.255.20, Flags: 0x00
      SPRING Algorithm - Algo: 0
      SRv6 Capability - Flags: 0
      Node MSD Advertisement Sub-TLV:Type:  23, Length: 10
      SRv6 Maximum Segments Left MSD:Type:  41, Value: 6
      SRv6 Maximum Pop MSD:Type:  42, Value: 7
      SRv6 Maximum Insert MSD:Type:  43, Value: 5
      SRv6 Maximum Encap MSD:Type:  44, Value: 6
      SRv6 Maximum End D MSD:Type:  45, Value: 6
      IPv6 TE Router Id: 2001:db8:20:255::20
    IS neighbor: R6.00, Internal, Metric: default 10
    IS neighbor: R3.00, Internal, Metric: default 10
    Extended IS Reachability TLV, Type: 22, Length: 174
    IS extended neighbor: R6.00, Metric: default 10 SubTLV len: 76
      IPv6 address: 2001:db8:67::2
      Neighbor's IP address: 192.168.255.6
      Neighbor's IPv6 address: 2001:db8:67::1
      Local interface index: 361, Remote interface index: 364
      P2P SRV6 END-X-SID:2001:db8:0:a7:1a76:: , Flags:B-P, Weight:0, Algorithm:0
       Flags:0xa0(B:1,S:0,P:1), Flavor: PSP
    IS extended neighbor: R3.00, Metric: default 10 SubTLV len: 76
      IPv6 address: 2001:db8:37::2
      Neighbor's IP address: 192.168.255.3
      Neighbor's IPv6 address: 2001:db8:37::1
```

```
      Local interface index: 360, Remote interface index: 336
      P2P SRV6 END-X-SID:2001:db8:0:a7:1a73:: , Flags:B-P, Weight:0, Algorithm:0
       Flags:0xa0(B:1,S:0,P:1), Flavor: PSP
  No queued transmissions
```

Meaning

The presence of `SRv6 SID: 2001:db8:0:a7:d71::` with `Flavor: USD` confirms that SRv6 is enabled with a SID decapsulate flavor on the R7 router. The output also shows that the interfaces at R7 have been configured for TI-LFA protection using a PSP flavor.

Verifying the Route to CE2 Uses an SRv6 Tunnel

Purpose

Display the route to the IPv4 subnet at R7 to confirm the next hop points to an SRv6 tunnel.

Action

From operational mode, run the **show route 172.16.20.0/24** command on router R0.

```
user@R0> show route 172.16.20.0/24

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

172.16.20.0/24     *[BGP/170] 05:20:58, localpref 100, from 2001:db8:2:255::2
                      AS path: I, validation-state: unverified
                       to fe80::2e6b:f5ff:fedb:e800 via xe-0/0/0:0.0, SRV6-Tunnel, Dest:
2001:db8:0:a7:d71::
                    >  to fe80::2e6b:f5ff:feb4:4000 via xe-0/0/0:2.0, SRV6-Tunnel, Dest:
2001:db8:0:a7:d71::
```

Meaning

The output confirms that R0 has learned the route to the `172.16.20.0/24` subnet through its BGP session to R2, which recall is configured as a route reflector in this example. The next hops confirm that an SRv6 tunnel to the R7 router has been installed for this route. Two next hops are available in keeping with their being two equal cost paths between the R0 and R7 routers in the example topology.

Test IPv4 Connectivity Between CE1 and CE2

Purpose

Generate pings to verify IPv4 connectivity between the CE devices over the IPv6 provider core.

Action

From operational mode, run the **ping 172.16.20.2 source 172.16.10.2 count 2** command on router R0.

```
user@R0> ping 172.16.20.2 source 172.16.10.2 count 2
PING 172.16.20.2 (172.16.20.2): 56 data bytes
64 bytes from 172.16.20.2: icmp_seq=0 ttl=64 time=114.922 ms
64 bytes from 172.16.20.2: icmp_seq=1 ttl=64 time=89.558 ms

--- 172.16.20.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 89.558/102.240/114.922/12.682 ms
```

Meaning

The output confirms IPv4 connectivity is working between the CE device networks. This provides verification that SRv6 tunneling over an IPv6 provider core is working properly in this example.

RELATED DOCUMENTATION

locator

srv6

SRv6 Network Programming and Layer 3 Services in BGP Networks

SUMMARY

IN THIS SECTION

Overview of SRv6 Network Programming and Layer 3 Services over SRv6 in BGP

IN THIS SECTION

Benefits of SRv6 Network Programming

- Flexible deployment—BGP leverages the segment routing capability of devices to set up Layer 3 VPN tunnels. SRv6 ingress node can transport IPv4 packets even if the transit routers are not SRv6-capable. This eliminates the need to deploy segment routing on all nodes in an IPv6 network.

- Seamless deployment—Network programming depends entirely on the IPv6 header and the header extension to transport a packet, eliminating the need for protocols such as MPLS. This ensures a seamless deployment without any major hardware or software upgrade in a core IPv6 network.
- Single-device versatility—Junos OS supports multiple functions on a single segment identifier (SID) and can inter-operate in the insert mode and the encapsulation mode. This allows a single device to simultaneously play the provider (P) router and the provider edge (PE) router roles.

SRv6 Network Programming in BGP Networks

Network programming is the capability of a network to encode a network program into individual instructions that are inserted into the IPv6 packet headers. The Segment Routing Header (SRH) is a type of IPv6 routing extension header that contains a segment list encoded as an SRv6 SID. An SRv6 SID consists of the locator, which is an IPv6 address, and a function that defines a particular task for each SRv6-capable node in the SRv6 network. SRv6 network programming eliminates the need for MPLS and provides flexibility to leverage segment routing.

NOTE: Ensure that you use a unique SID, which BGP uses to allocate an SRv6 SID.

To configure IPv4 transport over the SRv6 core, include the `end-dt4-sid` *`sid`* statement at the `[edit protocols bgp source-packet-routing srv6 locator` *`name`*`]` hierarchy level.

To configure IPv6 transport over the SRv6 core, include the `end-dt6-sid` *`sid`* statement at the `[edit routing protocols bgp source-packet-routing srv6 locator` *`name`*`]` hierarchy level.

To configure IPv4 and IPv6 transport over the SRv6 core, include the `end-dt46-sid` *`sid`* statement at the `[edit routing protocols bgp source-packet-routing srv6 locator` *`name`*`]` hierarchy level. The end-dt4-sid statement denotes the endpoint SID with de-encapsulation and IPv4 table lookup. The end dt6-sid statement is the endpoint with de-encapsulation and IPv6 table lookup. The end-dt46-sid statement is the endpoint with decapsulation and specific IP table lookup. The end-dt46 is a variant of end.dt4 and end.dt6 behavior. BGP allocates these values for IPv4 and IPv6 Layer3 VPN service SIDs.

Layer 3 VPN Services over the SRv6 Core

When connecting to the egress PE, the ingress PE encapsulates the payload in an outer IPv6 header where the destination address is the SRv6 service SID associated with the related BGP route update. The egress PE sets the next hop to one of its IPv6 addresses that is also the SRv6 locator from which the SRv6 service SID is allocated. Multiple routes can resolve through the same segment routing policy.

Figure 33: SRv6 Packet Encapsulation

You can configure BGP-based Layer 3 service over the SRv6 core. You can enable Layer 3 overlay services with BGP as the control plane and SRv6 as the dataplane. SRv6 network programming provides flexibility to leverage segment routing without deploying MPLS. Such networks depend only on the IPv6 headers and header extensions for transmitting data.

NOTE: Ensure that the `end-dt4-sid` *sid* and the `end-dt6-sid` *sid* are the last SIDs in the segment list, or the destination address of the packet with no SRH header.

To configure IPv4 VPN services over the SRv6 core, include the `end-dt4-sid` statement at the `[edit routing-instances` *instance-name* `protocols bgp source-packet-routing srv6 locator` *name*`]` hierarchy level.

The end dt46 SID must be the last segment in a segment routing policy, and a SID instance must be associated with an IPv4 FIB table and an IPv6 FIB table.

Advertising Layer 3 VPN Services to BGP Peers

BGP advertises the reachability of prefixes of a particular service from an egress PE device to ingress PE nodes. BGP messages exchanged between PE devices carry SRv6 service SIDs, which BGP uses to interconnect PE devices to form VPN sessions. For Layer 3 VPN services where BGP uses a per-VRF SID allocation, the same SID is shared across multiple network layer reachability information (NLRI) address families.

To advertise SRv6 services to BGP peers at the egress node, include the `advertise-srv6-service` statement at the `[edit protocols bgp family inet6-vpn unicast]` hierarchy level.

Egress PE devices that support SRv6-based Layer 3 services advertise overlay service prefixes along with a service SID. The BGP ingress node receives these advertisements and adds the prefix to the corresponding virtual routing and forwarding (VRF) table.

To accept SRv6 services at the ingress node, include the `accept-srv6-service` statement at the `[edit protocols bgp family inet6-vpn unicast]` hierarchy level.

Supported and Unsupported Features for SRv6 Network Programming in BGP

Junos OS supports the following features with SRv6 Network Programming in BGP:

- Ingress devices support seven SIDs in the reduced mode including the VPN SID

- Egress devices support seven SIDs including the VPN SID
- Endpoint with de-encapsulation and specific IP table lookup (End.DT46 SID)
- VPN options C

Junos OS does not support the following features in conjunction with SRv6 Network Programming in BGP:

- Fragmentation and reassembly in SRv6 tunnels
- VPN options B

SEE ALSO

srv6

advertise-srv6-service

accept-srv6-service

Example: Configuring Layer 3 Services over SRv6 in BGP Networks

IN THIS SECTION

This example shows how to configure SRv6 network programming and Layer 3 VPN services in BGP Networks. SRv6 network programming provides flexibility to leverage segment routing without deploying MPLS. This feature is useful for service providers whose networks are predominantly IPv6 and have not deployed MPLS.

Requirements

This example uses the following hardware and software components:

- Five MX Series routers with MPC7E, MPC8E, or MPC9E line cards
- Junos OS Release 20.4R1 or later

Overview

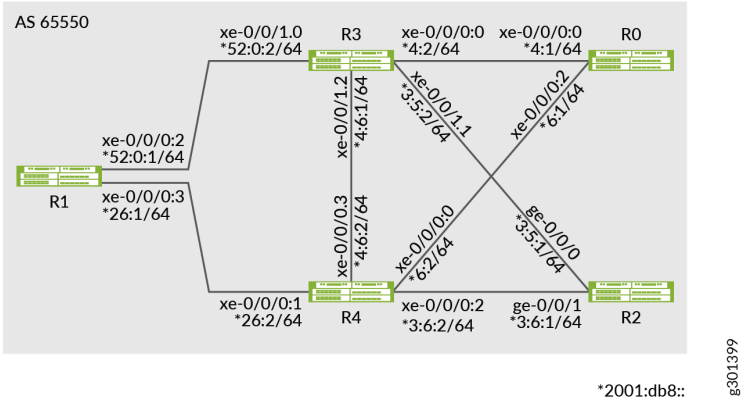
IN THIS SECTION

You can configure BGP-based Layer 3 services over the SRv6 core network. With SRv6 network programming, networks depend only on the IPv6 headers and header extensions for transmitting data. You can enable Layer 3 overlay services with BGP as the control plane and SRv6 as the dataplane.

Topology

In Figure 34 on page 434, Router R0 is the ingress and Router R1 and R2 are the egress routers that support IPv4-only customer edge devices. Routers R3 and R4 comprise an IPv6-only provider core network. All routers belong to the same autonomous system. IS-IS is the interior gateway protocol configured to support SRv6 in the IPv6 core routers R3 and R4. In this example, BGP is configured on routers R0, R1, and R2. Router R0 is configured as an IPv6 route reflector with IBGP peering sessions to both Router R1 and Router R2. The egress Router R1 advertises the L3VPN SID to ingress Router R0, which accepts and updates the VRF table.

Figure 34: Layer 3 Services over SRv6 in BGP Networks



From R1, BGP routes are advertised with next-hop self to Router R0. Router R0 has two paths to R1, the primary path through R3 and the backup path through R4. In Router R0 , the primary path is with default metric and the backup path is configured with metric 50. Here are some of the routes that are advertised from Router R1 to R0:

IPv4	21.0.0.0
IPv6	2001:21::
IPv4 VPN	31.0.0.0
IPv6 VPN	2001:31::

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter commit from configuration mode.

Router R0

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/0:0 unit 0 family inet address 1.4.1.1/30
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8::4:1/64

set interfaces xe-0/0/0:2 unit 0 family inet address 1.6.1.1/30
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8::6:1/64
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::0/128
set policy-options policy-statement adv_global term v4 from route-filter 20.0.0.0/8 orlonger
```

```
set policy-options policy-statement adv_global term v4 then next-hop self
set policy-options policy-statement adv_global term v4 then accept
set policy-options policy-statement adv_global term v6 from route-filter 2001:20::/64 orlonger
set policy-options policy-statement adv_global term v6 then next-hop self
set policy-options policy-statement adv_global term v6 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options community vpn1-target members target:100:1
set policy-options community vpn2-target members target:100:2
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 local-address 11.1.1.5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 peer-as 1002
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 neighbor 11.1.1.6
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 local-address 2001:11:1:1::5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 peer-as 1002
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 neighbor 2001:11:1:1::6
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid
3001::4
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid
3001::5
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface xe-0/0/0:3.1
set routing-instances vpn1 route-distinguisher 100:1
set routing-instances vpn1 vrf-target target:100:1
set routing-options source-packet-routing srv6 locator loc1 3001::/64
set routing-options source-packet-routing srv6 no-reduced-srh
set routing-options router-id 128.53.38.52
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols bgp group to-PE-all type internal
set protocols bgp group to-PE-all local-address abcd::128:53:38:52
set protocols bgp group to-PE-all family inet unicast extended-nexthop
set protocols bgp group to-PE-all family inet unicast advertise-srv6-service
set protocols bgp group to-PE-all family inet unicast accept-srv6-service
set protocols bgp group to-PE-all family inet-vpn unicast extended-nexthop
set protocols bgp group to-PE-all family inet-vpn unicast advertise-srv6-service
set protocols bgp group to-PE-all family inet-vpn unicast accept-srv6-service
set protocols bgp group to-PE-all family inet6 unicast advertise-srv6-service
set protocols bgp group to-PE-all family inet6 unicast accept-srv6-service
set protocols bgp group to-PE-all family inet6-vpn unicast advertise-srv6-service
```

```
set protocols bgp group to-PE-all family inet6-vpn unicast accept-srv6-service
set protocols bgp group to-PE-all export adv_global
set protocols bgp group to-PE-all cluster 128.53.38.52
set protocols bgp group to-PE-all neighbor abcd::128:53:35:39
set protocols bgp group to-PE-all neighbor abcd::128:53:35:35
set protocols bgp group to-TG-global-v4 type external
set protocols bgp group to-TG-global-v4 local-address 11.1.1.1
set protocols bgp group to-TG-global-v4 family inet unicast
set protocols bgp group to-TG-global-v4 family inet6 unicast
set protocols bgp group to-TG-global-v4 peer-as 1001
set protocols bgp group to-TG-global-v4 neighbor 11.1.1.2
set protocols bgp group to-TG-global-v6 type external
set protocols bgp group to-TG-global-v6 local-address 2001:11:1:1::1
set protocols bgp group to-TG-global-v6 family inet6 unicast
set protocols bgp group to-TG-global-v6 peer-as 1001
set protocols bgp group to-TG-global-v6 neighbor 2001:11:1:1::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid 3001::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid 3001::3
set protocols isis interface all
set protocols isis interface fxp0.0 disable

set protocols isis level 1 disable
```

Router R1

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/0:2 unit 0 family inet address 2.5.1.1/30
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8::52:0:1/64
set interfaces xe-0/0/0:3 unit 0 family inet address 2.6.1.1/30
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8::26:1/64
set policy-options policy-statement adv_global term v4 from route-filter 21.0.0.0/8 orlonger
set policy-options policy-statement adv_global term v4 from route-filter 12.1.1.1/30 orlonger
set policy-options policy-statement adv_global term v4 then next-hop self
set policy-options policy-statement adv_global term v4 then accept
set policy-options policy-statement adv_global term v6 from route-filter 2001:21::/64 orlonger
set policy-options policy-statement adv_global term v6 from route-filter 2001:12:1:1::1/126
orlonger
set policy-options policy-statement adv_global term v6 then next-hop self
set policy-options policy-statement adv_global term v6 then accept
set policy-options policy-statement adv_vpn1 term v4 from route-filter 31.0.0.0/8 orlonger
```

```
set policy-options policy-statement adv_vpn1 term v4 from route-filter 12.1.1.5/30 orlonger
set policy-options policy-statement adv_vpn1 term v4 then community set vpn1-target
set policy-options policy-statement adv_vpn1 term v4 then next-hop self
set policy-options policy-statement adv_vpn1 term v4 then accept
set policy-options policy-statement adv_vpn1 term v6 from route-filter 2001:31::/64 orlonger
set policy-options policy-statement adv_vpn1 term v6 from route-filter 2001:12:1:1::5/126
orlonger
set policy-options policy-statement adv_vpn1 term v6 then community set vpn1-target
set policy-options policy-statement adv_vpn1 term v6 then next-hop self
set policy-options policy-statement adv_vpn1 term v6 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options community vpn1-target members target:100:1
set policy-options community vpn2-target members target:100:2
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 local-address 12.1.1.5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 peer-as 1012
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 neighbor 12.1.1.6
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 local-address 2001:12:1:1::5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 peer-as 1012
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 neighbor 2001:12:1:1::6
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid
3011::4
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid
3011::5
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface xe-0/0/1:0.1
set routing-instances vpn1 route-distinguisher 100:1
set routing-instances vpn1 vrf-export adv_vpn1
set routing-instances vpn1 vrf-target target:100:1
set routing-options source-packet-routing srv6 locator loc1 3011::/64
set routing-options source-packet-routing srv6 no-reduced-srh
set routing-options rib inet6.3 static route abcd::128:53:38:52/128 next-hop self
set routing-options rib inet6.3 static route abcd::128:53:38:52/128 resolve
set routing-options rib inet6.0 static route abcd::128:53:38:52/128 next-hop self
set routing-options rib inet6.0 static route abcd::128:53:38:52/128 resolve
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address abcd::128:53:35:39
```

```
set protocols bgp group to-RR family inet unicast extended-nexthop
set protocols bgp group to-RR family inet unicast advertise-srv6-service
set protocols bgp group to-RR family inet unicast accept-srv6-service
set protocols bgp group to-RR family inet-vpn unicast extended-nexthop
set protocols bgp group to-RR family inet-vpn unicast advertise-srv6-service
set protocols bgp group to-RR family inet-vpn unicast accept-srv6-service
set protocols bgp group to-RR family inet6 unicast advertise-srv6-service
set protocols bgp group to-RR family inet6 unicast accept-srv6-service
set protocols bgp group to-RR family inet6-vpn unicast advertise-srv6-service
set protocols bgp group to-RR family inet6-vpn unicast accept-srv6-service
set protocols bgp group to-RR export adv_global
set protocols bgp group to-RR neighbor abcd::128:53:38:52
set protocols bgp group to-TG-global-v4 type external
set protocols bgp group to-TG-global-v4 local-address 12.1.1.1
set protocols bgp group to-TG-global-v4 family inet unicast
set protocols bgp group to-TG-global-v4 family inet6 unicast
set protocols bgp group to-TG-global-v4 peer-as 1011
set protocols bgp group to-TG-global-v4 neighbor 12.1.1.2
set protocols bgp group to-TG-global-v6 type external
set protocols bgp group to-TG-global-v6 local-address 2001:12:1:1::1
set protocols bgp group to-TG-global-v6 family inet6 unicast
set protocols bgp group to-TG-global-v6 peer-as 1011
set protocols bgp group to-TG-global-v6 neighbor 2001:12:1:1::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid 3011::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid 3011::3
set protocols isis interface all
set protocols isis interface fxp0.0 disable

set protocols isis level 1 disable
```

Router R2

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 3.5.1.1/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8::3:5:1/64
set interfaces ge-0/0/1 unit 0 family inet address 3.6.1.1/30
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8::3:6:1/64
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::2/128
set policy-options policy-statement adv_global term v4 from route-filter 22.0.0.0/8 orlonger
set policy-options policy-statement adv_global term v4 from route-filter 13.1.1.1/30 orlonger
```

```
set policy-options policy-statement adv_global term v4 then next-hop self
set policy-options policy-statement adv_global term v4 then accept
set policy-options policy-statement adv_global term v6 from route-filter 2001:22::/64 orlonger
set policy-options policy-statement adv_global term v6 from route-filter 2001:13:1:1::1/126
orlonger
set policy-options policy-statement adv_global term v6 then next-hop self
set policy-options policy-statement adv_global term v6 then accept
set policy-options policy-statement adv_vpn1 term v4 from route-filter 32.0.0.0/8 orlonger
set policy-options policy-statement adv_vpn1 term v4 from route-filter 13.1.1.5/30 orlonger
set policy-options policy-statement adv_vpn1 term v4 then community set vpn1-target
set policy-options policy-statement adv_vpn1 term v4 then next-hop self
set policy-options policy-statement adv_vpn1 term v4 then accept
set policy-options policy-statement adv_vpn1 term v6 from route-filter 2001:32::/64 orlonger
set policy-options policy-statement adv_vpn1 term v6 from route-filter 2001:13:1:1::5/126
orlonger
set policy-options policy-statement adv_vpn1 term v6 then community set vpn1-target
set policy-options policy-statement adv_vpn1 term v6 then next-hop self
set policy-options policy-statement adv_vpn1 term v6 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options community vpn1-target members target:100:1
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 local-address 13.1.1.5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 peer-as 1022
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 neighbor 13.1.1.6
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 local-address 2001:13:1:1::5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 peer-as 1022
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 neighbor 2001:13:1:1::6
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid
3021::4
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid
3021::5
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-0/0/2.1
set routing-instances vpn1 route-distinguisher 100:1
set routing-instances vpn1 vrf-export adv_vpn1
set routing-instances vpn1 vrf-target target:100:1
set routing-options source-packet-routing srv6 locator loc1 3021::/64
set routing-options source-packet-routing srv6 no-reduced-srh
set routing-options rib inet6.3 static route abcd::128:53:38:52/128 next-hop self
```

```
set routing-options rib inet6.3 static route abcd::128:53:38:52/128 resolve
set routing-options rib inet6.0 static route abcd::128:53:38:52/128 next-hop self
set routing-options rib inet6.0 static route abcd::128:53:38:52/128 resolve
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address abcd::128:53:35:35
set protocols bgp group to-RR family inet unicast extended-nexthop
set protocols bgp group to-RR family inet unicast advertise-srv6-service
set protocols bgp group to-RR family inet unicast accept-srv6-service
set protocols bgp group to-RR family inet-vpn unicast extended-nexthop
set protocols bgp group to-RR family inet-vpn unicast advertise-srv6-service
set protocols bgp group to-RR family inet-vpn unicast accept-srv6-service
set protocols bgp group to-RR family inet6 unicast advertise-srv6-service
set protocols bgp group to-RR family inet6 unicast accept-srv6-service
set protocols bgp group to-RR family inet6-vpn unicast advertise-srv6-service
set protocols bgp group to-RR family inet6-vpn unicast accept-srv6-service
set protocols bgp group to-RR export adv_global
set protocols bgp group to-RR neighbor abcd::128:53:38:52
set protocols bgp group to-TG-global-v4 type external
set protocols bgp group to-TG-global-v4 local-address 13.1.1.1
set protocols bgp group to-TG-global-v4 family inet unicast
set protocols bgp group to-TG-global-v4 family inet6 unicast
set protocols bgp group to-TG-global-v4 peer-as 1021
set protocols bgp group to-TG-global-v4 neighbor 13.1.1.2
set protocols bgp group to-TG-global-v6 type external
set protocols bgp group to-TG-global-v6 local-address 2001:13:1:1::1
set protocols bgp group to-TG-global-v6 family inet6 unicast
set protocols bgp group to-TG-global-v6 peer-as 1021
set protocols bgp group to-TG-global-v6 neighbor 2001:13:1:1::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid 3021::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid 3021::3
set protocols isis interface all
set protocols isis interface fxp0.0 disable

set protocols isis level 1 disable
```

Router R3

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/0:0 unit 0 family inet address 1.4.1.2/30
set interfaces xe-0/0/0:0 unit 0 family iso
```

```
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8::4:2/64
set interfaces xe-0/0/1:0 unit 0 family inet address 2.5.1.2/30
set interfaces xe-0/0/1:0 unit 0 family iso
set interfaces xe-0/0/1:0 unit 0 family inet6 address 2001:db8::52:0:2/64
set interfaces xe-0/0/1:1 unit 0 family inet address 3.5.1.2/30
set interfaces xe-0/0/1:1 unit 0 family iso
set interfaces xe-0/0/1:1 unit 0 family inet6 address 2001:db8::3:5:2/64
set interfaces xe-0/0/1:2 unit 0 family inet address 4.6.1.1/30
set interfaces xe-0/0/1:2 unit 0 family iso
set interfaces xe-0/0/1:2 unit 0 family inet6 address 2001:db8::4:6:1/64
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::3/128
set routing-options autonomous-system 100
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis level 1 disable
```

Router R4

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/0:0 unit 0 family inet address 1.6.1.2/30
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8::6:2/64
set interfaces xe-0/0/0:1 unit 0 family inet address 2.6.1.2/30
set interfaces xe-0/0/0:1 unit 0 family iso
set interfaces xe-0/0/0:1 unit 0 family inet6 address 2001:db8::26:2/64
set interfaces xe-0/0/0:2 unit 0 family inet address 3.6.1.2/30
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8::3:6:2/64
set interfaces xe-0/0/0:3 unit 0 family inet address 4.6.1.2/30
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8::4:6:2/64
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::4/128
set routing-options autonomous-system 100
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis level 1 disable
```

Configure Router R0

Step-by-Step Procedure

To configure SRv6 network programming with Layer 3 VPN services, perform the following steps on Router R0:

1. Configure the device interfaces to enable IP transport.

```
[edit]
user@R0# set interfaces xe-0/0/0:0 unit 0 family inet address 1.4.1.1/30
user@R0# set interfaces xe-0/0/0:0 unit 0 family iso
user@R0# set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8::4:1/64

user@R0# set interfaces xe-0/0/0:2 unit 0 family inet address 1.6.1.1/30
user@R0# set interfaces xe-0/0/0:2 unit 0 family iso
user@R0# set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8::6:1/64
```

2. Configure the router ID and autonomous system (AS) number to propagate routing information within a set of routing devices that belong to the same AS.

```
[edit]
user@R0# set routing-options router-id 128.53.38.52
user@R0# set routing-options autonomous-system 100
```

3. Enable SRv6 globally and the locator address to indicate the SRv6 capability of the router. SRv6 SID is an IPv6 address that consists of the locator and a function. The routing protocols advertise the locator addresses.

```
[edit]
user@R0# set routing-options source-packet-routing srv6 locator loc1 3001::/64
user@R0# set routing-options source-packet-routing srv6 no-reduced-srh
```

4. Configure an external routing instance VPN1 for both IPv4 and IPv6 traffic. Configure the BGP protocol for VPN1 to enable peering and traffic transport between the provider edge devices.

```
[edit]
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 type external
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 local-address 11.1.1.5
```

```
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet unicast
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet6 unicast
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 peer-as 1002
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 neighbor 11.1.1.6
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 type external
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 local-address
2001:11:1:1::5
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 family inet6 unicast
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 peer-as 1002
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 neighbor
2001:11:1:1::6
```

5. Configure the VPN type and a unique route distinguisher for each PE router participating in the routing instance.

```
[edit]
user@R0# set routing-instances vpn1 instance-type vrf
user@R0# set routing-instances vpn1 interface xe-0/0/0:3.1
user@R0# set routing-instances vpn1 route-distinguisher 100:1
user@R0# set routing-instances vpn1 vrf-target target:100:1
```

6. Configure the end-dt4 and end-dt6 SID values for enabling the Layer 3 VPN services.

```
[edit]
user@R0# set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1
end-dt4-sid 3001::4
user@R0# set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1
end-dt6-sid 3001::5
```

7. Define a policy to load-balance packets.

```
[edit]
user@R0# set policy-options policy-statement pplb then load-balance per-packet
user@R0# set policy-options community vpn1-target members target:100:1
user@R0# set policy-options community vpn2-target members target:100:2
```

8. Apply the per-packet policy to enable load balancing of traffic.

```
[edit]
user@R0# set routing-options forwarding-table export pplb
```

9. Define a policy adv_global to accept routes advertised from R1.

```
[edit]
user@R0# set policy-options policy-statement adv_global term v4 from route-filter
20.0.0.0/8 orlonger
user@R0# set policy-options policy-statement adv_global term v4 then next-hop self
user@R0# set policy-options policy-statement adv_global term v4 then accept
user@R0# set policy-options policy-statement adv_global term v6 from route-filter
2001:20::/64 orlonger
user@R0# set policy-options policy-statement adv_global term v6 then next-hop self
user@R0# set policy-options policy-statement adv_global term v6 then accept
```

10. Configure BGP on the core-facing interface to establish internal and external peering sessions.

```
[edit]
user@R0# set protocols bgp group to-PE-all type internal
user@R0# set protocols bgp group to-PE-all local-address abcd::128:53:38:52
user@R0# set protocols bgp group to-PE-all family inet unicast extended-nexthop
user@R0# set protocols bgp group to-PE-all family inet unicast advertise-srv6-service
user@R0# set protocols bgp group to-PE-all family inet unicast accept-srv6-service
user@R0# set protocols bgp group to-PE-all family inet-vpn unicast extended-nexthop
user@R0# set protocols bgp group to-PE-all export adv_global
user@R0# set protocols bgp group to-PE-all cluster 128.53.38.52
user@R0# set protocols bgp group to-PE-all neighbor abcd::128:53:35:39
user@R0# set protocols bgp group to-PE-all neighbor abcd::128:53:35:35
user@R0# set protocols bgp group to-TG-global-v4 type external
user@R0# set protocols bgp group to-TG-global-v4 local-address 11.1.1.1
user@R0# set protocols bgp group to-TG-global-v4 family inet unicast
user@R0# set protocols bgp group to-TG-global-v4 family inet6 unicast
user@R0# set protocols bgp group to-TG-global-v4
user@R0# set protocols bgp group to-TG-global-v4 neighbor 11.1.1.2
user@R0# set protocols bgp group to-TG-global-v6 type external
user@R0# set protocols bgp group to-TG-global-v6 local-address 2001:11:1:1::1
user@R0# set protocols bgp group to-TG-global-v6 family inet6 unicast
```

```
user@R0# set protocols bgp group to-TG-global-v6 peer-as 1001
user@R0# set protocols bgp group to-TG-global-v6 neighbor 2001:11:1:1::2
```

11. Enable the device to advertise the SRv6 services to BGP peers and to accept the routes advertised by the egress provider edge (PE) devices.

```
[edit]
user@R0# set protocols bgp group to-PE-all family inet-vpn unicast advertise-srv6-service
user@R0# set protocols bgp group to-PE-all family inet-vpn unicast accept-srv6-service
user@R0# set protocols bgp group to-PE-all family inet6 unicast advertise-srv6-service
user@R0# set protocols bgp group to-PE-all family inet6 unicast accept-srv6-service
user@R0# set protocols bgp group to-PE-all family inet6-vpn unicast advertise-srv6-service
user@R0# set protocols bgp group to-PE-all family inet6-vpn unicast accept-srv6-service
```

12. Enable IS-IS as the interior gateway protocol (IGP) for routing traffic between the core provider routers.

```
[edit]
user@R0# set protocols isis interface all
user@R0# set protocols isis interface fxp0.0 disable
user@R0#
user@R0# set protocols isis level 1 disable
```

13. Configure the end-dt4 and end-dt6 SID value for the prefix segments. End-dt4 is the endpoint SID with decapsulation and IPv4 table lookup and end-dt6 is the endpoint with decapsulation and IPv6 table lookup. BGP allocates these for IPv4 and IPv6 Layer3 VPN services SIDs.

```
[edit]
user@R0# set protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid 3001::2
user@R0# set protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid 3001::3
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@R0# show interfaces
xe-0/0/0:0 {
    unit 0 {
        family inet {
            address 1.4.1.1/30;
        }
        family iso;
        family inet6 {
            address 2001:db8::4:1/64;
        }
    }
}
xe-0/0/0:1 {
    unit 0 {
        family inet {
            address 1.5.1.1/30;
        }
        family iso;
        family inet6 {
            address 2001:1:4:2::1/126;
        }
    }
}
xe-0/0/0:2 {
    unit 0 {
        family inet {
            address 1.6.1.1/30;
        }
        family iso;
        family inet6 {
            address 2001:db8::6:1/64;
        }
```

```
    }
}
```

```
[edit]
user@R0# show protocols
bgp {
    group to-PE-all {
        type internal;
        local-address abcd::128:53:38:52;
        family inet {
            unicast {
                extended-nexthop;
                advertise-srv6-service;
                accept-srv6-service;
            }
        }
        family inet-vpn {
            unicast {
                extended-nexthop;
                advertise-srv6-service;
                accept-srv6-service;
            }
        }
        family inet6 {
            unicast {
                advertise-srv6-service;
                accept-srv6-service;
            }
        }
        family inet6-vpn {
            unicast {
                advertise-srv6-service;
                accept-srv6-service;
            }
        }
        export adv_global;
        cluster 128.53.38.52;
        neighbor abcd::128:53:35:39;
        neighbor abcd::128:53:35:35;
    }
    group to-TG-global-v4 {
```

```
        type external;
        local-address 11.1.1.1;
        family inet {
            unicast;
        }
        family inet6 {
            unicast;
        }
        peer-as 1001;
        neighbor 11.1.1.2;
    }
    group to-TG-global-v6 {
        type external;
        local-address 2001:11:1:1::1;
        family inet6 {
            unicast;
        }
        peer-as 1001;
        neighbor 2001:11:1:1::2;
    }
    source-packet-routing {
        srv6 {
            locator loc1 {
                end-dt4-sid 3001::2;
                end-dt6-sid 3001::3;
            }
        }
    }
}
isis {
    interface all;
    interface fxp0.0 {
        disable;
    }

    level 1 disable;
}
```

```
[edit]
user@R0# show policy-options
policy-options {
```

```
    policy-statement adv_global {
        term v4 {
            from {
                route-filter 20.0.0.0/8 orlonger;
            }
            then {
                next-hop self;
                accept;
            }
        }
        term v6 {
            from {
                route-filter 2001:20::/64 orlonger;
            }
            then {
                next-hop self;
                accept;
            }
        }
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
    community vpn1-target members target:100:1;
    community vpn2-target members target:100:2;
}
```

```
[edit]
user@R0# show routing-options
routing-options {
    source-packet-routing {
        srv6 {
            locator loc1 3001::/64;
            no-reduced-srh;
        }
    }

    router-id 128.53.38.52;
    autonomous-system 100;
```

```
    forwarding-table {
        export pplb;
    }
}
```

```
[edit]
user@R0# show routing-instances
routing-instances {
    vpn1 {
        protocols {
            bgp {
                group to-TG-vpn1-v4 {
                    type external;
                    local-address 11.1.1.5;
                    family inet {
                        unicast;
                    }
                    family inet6 {
                        unicast;
                    }
                    peer-as 1002;
                    neighbor 11.1.1.6;
                }
                group to-TG-vpn1-v6 {
                    type external;
                    local-address 2001:11:1:1::5;
                    family inet6 {
                        unicast;
                    }
                    peer-as 1002;
                    neighbor 2001:11:1:1::6;
                }
                source-packet-routing {
                    srv6 {
                        locator loc1 {
                            end-dt4-sid 3001::4;
                            end-dt6-sid 3001::5;
                        }
                    }
                }
            }
```

```
        }
        instance-type vrf;
        interface xe-0/0/0:3.1;
        route-distinguisher 100:1;
        vrf-target target:100:1;
    }
}
```

When done configuring the device, enter `commit` from the configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verify that the advertised IPv4 route is installed in the IPv4 table

Purpose

Verify that ingress router R0 has learned the route to the IPv4 prefix 21.0.0.0 from the egress router R1.

Action

From operational mode, run the **show route 21.0.0.0** command on router R0.

```
user@R0> show route 21.0.0.0
inet.0: 59 destinations, 59 routes (59 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

21.0.0.0/30        *[BGP/170] 09:15:25, localpref 100, from abcd::128:53:37:72
                      AS path: {65501} I, validation-state: unverified
                    >  to fe80::2e6b:f5ff:fe28:2bcb via ae0.0, SRV6-Tunnel, Dest: 3011::
```

```
                        to fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0, SRV6-Tunnel, Dest: 3011::
                        to fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0, SRV6-Tunnel, Dest: 3011::
```

Meaning

The output confirms that the IPv4 prefix 21.0.0.0 is installed in the inet.0 table.

Verify that SRv6 SID is installed in the IPv4 Table

Purpose

Verify that ingress Router R0 has received and accepted the SRv6 end-dt4 SID 3011::2 from the egress Router R1.

Action

From operational mode, run the **show route 21.0.0.0 extensive** command on Router R0.

```
user@> show route 21.0.0.0 extensive
inet.0: 59 destinations, 59 routes (59 active, 0 holddown, 0 hidden)
21.0.0.0/30 (1 entry, 1 announced)
TSI:
KRT in-kernel 21.0.0.0/30 -> {composite(716)}
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xc5aa39c
                Next-hop reference count: 20
                Source: abcd::128:53:37:72
                Next hop type: List, Next hop index: 1048574
        Next hop: ELNH Address 0xc5a9e88, selected
            Next hop type: Chain, Next hop index: 725
            Address: 0xc5a9e88
            Next-hop reference count: 1
            Next hop: ELNH Address 0xc5a9aa0
            SRV6-Tunnel: Reduced-SRH Encap-mode
             Src: abcd::128:53:35:39 Dest: 3011::
             Segment-list[0] 3011::
                Next hop type: Router, Next hop index: 700
                Address: 0xc5a9aa0
                Next-hop reference count: 4
                Next hop: fe80::2e6b:f5ff:fe28:2bcb via ae0.0
```

```
          Next hop: ELNH Address 0xc5a9eec
              Next hop type: Chain, Next hop index: 726
              Address: 0xc5a9eec
              Next-hop reference count: 1
              Next hop: ELNH Address 0xc5a9c30
              SRV6-Tunnel: Reduced-SRH Encap-mode
               Src: abcd::128:53:35:39 Dest: 3011::
               Segment-list[0] 3011::
                  Next hop type: Router, Next hop index: 702
                  Address: 0xc5a9c30
                  Next-hop reference count: 4
                  Next hop: fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0
          Next hop: ELNH Address 0xc5aa0e0
              Next hop type: Chain, Next hop index: 727
              Address: 0xc5aa0e0
              Next-hop reference count: 1
              Next hop: ELNH Address 0xc5a9780
              SRV6-Tunnel: Reduced-SRH Encap-mode
               Src: abcd::128:53:35:39 Dest: 3011::
               Segment-list[0] 3011::
                  Next hop type: Router, Next hop index: 647
                  Address: 0xc5a9780
                  Next-hop reference count: 20
                  Next hop: fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0
                  Protocol next hop: abcd::128:53:37:72
                  Composite next hop: 0xbd4e7d0 716 INH Session ID: 0x151
                  Indirect next hop: 0xc762204 1048582 INH Session ID: 0x151
                  State: <Active int Ext>
                  Local AS:   100 Peer AS:   100
                  Age: 9:13:44    Metric2: 20
                  Validation State: unverified
                  ORR Generation-ID: 0
                  Task: BGP_100.abcd::128:53:37:72
                  Announcement bits (1): 0-KRT
                  AS path: {65501}
                  Accepted
                  SRv6 SID: 3011::2
                  Localpref: 100
                  Router ID: 128.53.37.72
                  Composite next hops: 1
                           Protocol next hop: abcd::128:53:37:72 Metric: 20
                           Composite next hop: 0xbd4e7d0 716 INH Session ID: 0x151
                           Indirect next hop: 0xc762204 1048582 INH Session ID: 0x151
```

```
                        Indirect path forwarding next hops: 3
                                Next hop type: List
                                Next hop: fe80::2e6b:f5ff:fe28:2bcb via ae0.0
                                Next hop: fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0
                                Next hop: fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0
                                abcd::128:53:37:72/128 Originating RIB: inet6.3
                                  Metric: 20 Node path count: 1
                                  Indirect next hops: 1
                                Protocol next hop: 3011:: Metric: 20
                                Inode flags: 0x206 path flags: 0x0
                                Path fnh link: 0xc3bf4c0 path inh link: 0x0
                                Indirect next hop: 0xc76cd04 - INH Session ID: 0x0
                                Indirect path forwarding next hops: 3
                                        Next hop type: List
                                        Next hop: fe80::2e6b:f5ff:fe28:2bcb via ae0.0
                                        Next hop: fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0
                                        Next hop: fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0
                                        3011:: Originating RIB: inet6.3
                                          Metric: 20 Node path count: 1
                                          Forwarding nexthops: 3
                                                Next hop type: List
                                                Next hop: fe80::2e6b:f5ff:fe28:2bcb via ae0.0
                                                Next hop: fe80::2e6b:f5ff:fe28:2b04 via
xe-0/0/0:2.0
                                                Next hop: fe80::2e6b:f5ff:fe73:1e01 via
xe-0/0/0:3.0
```

Meaning

The output displays the SRv6 SID and confirms that an SRv6 tunnel is established between Routers R0 and R1.

Verify that the IPv6 VPN route is installed in the VPN table

Purpose

Verify that ingress router R0 has learned the route to the VPN IPv6 prefix 2001::30::/126 from the egress router R1.

Action

From operational mode, run the **show route 2001:31::** command on router R0.

```
user@R0> show route 2001:31::
vpn1.inet6.0: 36 destinations, 36 routes (36 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:31::/126      *[BGP/170] 09:15:40, localpref 100, from abcd::128:53:37:72
                      AS path: {65502} I, validation-state: unverified
                    >  to fe80::2e6b:f5ff:fe28:2bcb via ae0.0, SRV6-Tunnel, Dest: 3011::
                       to fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0, SRV6-Tunnel, Dest: 3011::
                       to fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0, SRV6-Tunnel, Dest: 3011::
```

Meaning

The output confirms that the route details for the prefix 2001:31::/126 are installed in the vpn.inet6.0 table.

Verify that the IPv4 VPN route is installed in the VPN table

Purpose

Verify that ingress router R0 has learned the route to the VPN IPv4 prefix 31.0.0.0 from the egress router R1.

Action

From operational mode, run the **show route 31.0.0.0** command on router R0.

```
user@R0> show route 31.0.0.0
vpn1.inet.0: 34 destinations, 34 routes (34 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

31.0.0.0/30        *[BGP/170] 09:15:29, localpref 100, from abcd::128:53:37:72
                      AS path: {65502} I, validation-state: unverified
                       to fe80::2e6b:f5ff:fe28:2bcb via ae0.0, SRV6-Tunnel, Dest: 3011::
                       to fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0, SRV6-Tunnel, Dest: 3011::
```

```
                    >  to fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0, SRV6-Tunnel, Dest: 3011::
```

Meaning

The output confirms that the IPv4 prefix 31.0.0.0 is installed in the vpn.inet.0 table.

SEE ALSO

- *srv6*
- *advertise-srv6-service*
- *accept-srv6-service*

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
change-completed	

Microloop Avoidance in SRv6 Networks

SUMMARY

IN THIS SECTION

Enable a post convergence path calculation on a device to avoid microloops if a link or metric change occurs in an SRv6 network. To configure microloop avoidance in an SRv6 network for both local and remote network events including link down, link-up, and metric-change, include the `microloop avoidance post-convergence-path delay` *`milliseconds`*statement at the `[edit protocols isis spf-options]` hierarchy level. For effective microloop avoidance, configure this feature on all the nodes in the network.

NOTE: Micro-loop avoidance is not a replacement for local repair mechanisms like TI-LFA which detects local failure very fast and activates a pre-computed loop-free-alternative path.

Benefits of Microloop Avoidance in SRv6

- Micro loop-free path avoids delays and traffic loss
- Microloop avoidance can prevent forwarding of looping packets and avoid wasteful bandwidth consumption
- Microloop avoidance path is computed only for the impacted links in case of multiple link failures. If the second link failure does not impact the computed microloop avoidance path, IS-IS continues to use the same microloop avoidance path.

RELATED DOCUMENTATION

IGP Microloop Avoidance | 105

EVPN E-LAN Overview

An Ethernet LAN (E-LAN), defined by the Metro Ethernet Forum (MEF), is a multipoint-to-multipoint transparent Layer 2 (L2) VLAN service that connects two or more user network interfaces (UNIs). The E-LAN provides full mesh connectivity for the UNI sites. A UNI is the dividing point between the responsibilities of the service provider and subscriber. Every UNI can communicate with any other UNI that is connected to the E-LAN service.

Ethernet VPN (EVPN) E-LAN is a framework for delivering multipoint-to-multipoint VPN service with the EVPN signaling mechanisms. E-LAN service allows service providers to offer services that manage the L2 learning very efficiently. In a multihoming scenario, the broadcast, unknown unicast, and multicast (BUM) traffic is handled by the provider edge (PE) device, acting as a designated forwarder (DF). The learned information is redistributed to other PEs in the network. The multihomed customer edge (CE) device connects a customer site to two or more PE devices providing redundant services.

The MEF standard has two different services for EVPN E-LAN:

- Ethernet Private LAN (EP-LAN), which offers a multipoint-to-multipoint ethernet virtual connection (EVC) between dedicated UNIs. EP-LAN is a port-based service.
- Ethernet Virtual Private LAN (EVP-LAN), which offers VLAN based service multiplexing, which means EVCs are paired per UNI.

EVPN E-LAN over SRv6

SUMMARY

This article shows the necessary steps to configure segment routing over IPv6 (SRv6), using an Ethernet VPN (EVPN) Ethernet LAN (E-LAN) network.

IN THIS SECTION

Overview

EVPN E-LAN serves as a framework for delivering multipoint-to-multipoint VPN service using EVPN signaling mechanisms. The egress provider edge (PE) device signals an SRv6 segment identifier (SID) with the VPN route. The ingress PE encapsulates the SRv6 SID in the VPN packet using an outer IPv6 header. The destination address is the SRv6 SID advertised by the egress PE, and is routable in the IPv6 underlay. The nodes between the PEs only need to support plain IPv6 forwarding. We support SRv6 micro-SID (uSID) and segment routing header (SRH) based control planes. Different endpoint behaviors are defined for SRv6 services on the egress node. See RFC8986 for information regarding the various endpoint behaviors.

Configure EVPN E-LAN

You can configure your EVPN routing instance following the steps below. You'll need to provide details specific to your implementation.

- Configure a MAC-VRF routing instance with the name of your choice.

```
set routing-instances <instance-name> instance-type mac-vrf
```

- Configure the E-LAN service. Here we're using `vlan-based`, however you can also use `vlan-aware` or `vlan-bundle`. If you require more than one VLAN, then `vlan-based` won't work.

```
set routing-instances <instance-name> service-type vlan-based
```

- Configure one or more VLAN IDs to fit your implementation.

```
set routing-instances <instance-name> vlan-id vlan
```

- Configure one or more VLANs to fit your implementation. You may configure more than one interface per VLAN, depending on your needs.

```
set routing-instances <instance-name> vlans <vlan-name> vlan-id <vlan>
set routing-instances <instance-name> vlans <vlan-name> interface <interface>
```

- Configure a unique RD and VRF target.

```
set routing-instances <instance-name> route-distinguisher <value>
set routing-instances <instance-name> vrf-target <target:x:x>
```

Configure SRv6

- Configure SRv6 encapsulation in EVPN.

```
set routing-instances <instance-name> protocols evpn encapsulation srv6
```

- Configure the SRv6 locator type in EVPN. You can choose either SRH (end-dt2-sid) or uSID (micro-dt2-sid).

```
set routing-instances <instance-name> protocols evpn source-packet-routing srv6 locator
<locator-name> (end-dt2-sid | micro-dt2-sid)
```

- Configure one or more locator blocks of various sizes. Juniper recommends a prefix length of 64. A format example, using private addressing, would look like **fd33:7ce6:27bc:168b::/64**.

```
set routing-options source-packet-routing srv6 locator <locator-name> <locator-block>
```

- Depending on the devices in your configuration, you should keep the following in mind:
  - (ACX Series) Only a 32-bit block size and a 16-bit uSID size are supported.
  - (ACX Series) The same locator configured under `[edit protocols isis source-packet-routing srv6 locator` *`locator-name`* `micro-node-sid]`, and under `[edit routing-instances` *`instance-name`* `mac-vrf protocols evpn source-packet-routing srv6 locator` *`locator-name`*`]` is not supported.
- Configure additional supporting statements for SRv6.

```
set chassis network-services enhanced-ip;

set interfaces lo0.0 family inet6 address <ipv6-address>;

set routing-options resolution preserve-nexthop-hierarchy;
```

- Check and commit your configuration.

RELATED DOCUMENTATION

dynamic-tunnels

multipath (Protocols BGP)

Configuring EVPN-VPWS over SRv6

EVPN VPWS provides point to point Layer 2 VPN service using EVPN signaling. EVPN-VPWS supports both single homed and multihomed (single-active or all-active) devices. EVPN-VPWS over SRv6 (Segment Routing over IPv6). SRv6 uses the IPv6 Segment Routing Header (SRH) extension to encode an order list of network instructions. The network instruction contains explicit information about SRv6 nodes that are available for packet processing on the path. The instruction also include task or function information for the SRv6 node in the SRv6 network. The SRH contains a list of 128-bit segment identifiers (SIDs) in the form of an IPv6 addresses. SIDs consist of the following:

- Locator—The locator is the first part of the SID and consists of the most significant bits. It represents the address of a particular SRv6 node. The locator is similar to a network address. It is used to route the packet.
- Function—The function is the second part of the SID. It defines the packet processing function that the node identified by the locator performs locally. Junos OS supports End.DX2 function for EVPN-VPWS. End.DX2 specifies endpoint decapsulation and L2 cross-connect behavior.

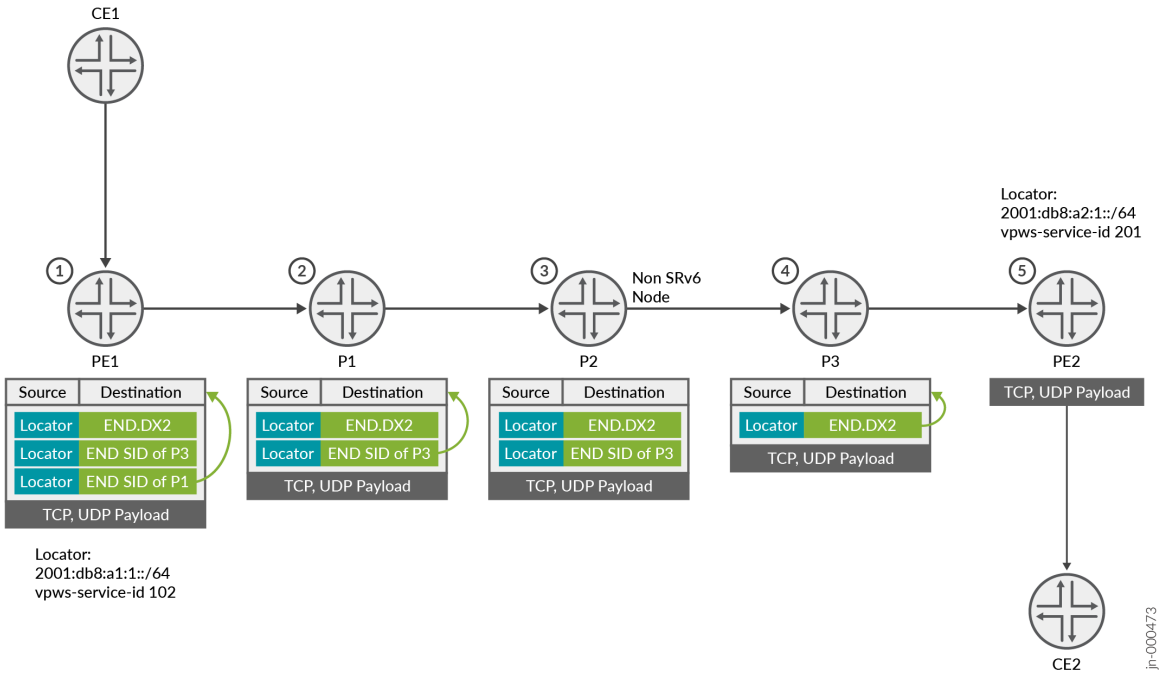
Benefits of EVPN-VPWS over SRv6

EVPN-VPWS over an SRv6 underlay has the following benefits IPv6 network:

1. Network Programming depends entirely on the IPv6 header and the header extension to transport a packet, eliminating protocols such as MPLS. This ensures a seamless deployment without any major hardware or software upgrade in a core IPv6 network.
2. Packets can be transported through an SRv6 ingress node even when the transit routers are not SRv6-capable. This eliminates the need to deploy segment routing on all nodes in an IPv6 network.

Figure 35 on page 463 illustrates how the SRH is processed by the nodes in a SRv6 topology .

Figure 35: SRH and Outer IPv6 Header Processing in a SRv6 Topology



1. PE1 encapsulates the payload with an SRH. The SRH list contains three SIDs. Each SID represent a SRv6 node along the segment path. The function on the last SID is END.DX2 endpoint.
2. P1 pops and processes the first SID at the bottom of the SRH list and copies the next SID to the outer destination. The SRH list contains 2 SIDS.
3. P2 is a non-SRv6 node. P2 forwards the packet on the current segment path with no further processing.
4. P3 pops and processes the second SID and copies the third SID in the SRH list.
5. PE2 pops and processes the third SID. END.DX2 identifies the CE facing interface and PE2 forwards the packet.

EVPN-VPWS builds upon an SRv6 baseline configuration. For more information about configuring SRv6, see Understanding SRv6 Network Programming and Layer 3 Services over SRv6 in BGP.

CLI Quick Configuration

To quickly configure EVPN-VPWS over SRv6, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

PE1

```
set chassis network-services enhanced-ip
set routing-instances EVPN-VPWS1 instance-type evpn-vpws
set routing-instances EVPN-VPWS1 protocols evpn encapsulation srv6
set routing-options source-packet-routing srv6 locator LOC1 2001:db8:a1:1::/64
set routing-options resolution preserve-nexthop-hierarchy
set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-id local 102
remote 201
set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-id source-
packet-routing srv6 locator LOC1 end-dx2-sid 2001:db8:a1:1:101::
set protocols bgp group IBGPv6 family evpn signaling advertise-srv6-service
set protocols bgp group IBGPv6 family evpn signaling accept-srv6-service
set routing-instances EVPN-VPWS1 route-distinguisher 65000:100
set routing-instances EVPN-VPWS1 vrf-target target:65000:200
set routing-instances EVPN-VPWS1 interface ge-0/0/1.1
```

PE2

```
set chassis network-services enhanced-ip
set routing-instances EVPN-VPWS1 instance-type evpn-vpws
set routing-instances EVPN-VPWS1 protocols evpn encapsulation srv6
set routing-options source-packet-routing srv6 locator LOC1 2001:db8:a1:2::/64
set routing-options resolution preserve-nexthop-hierarchy
set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-id local 201
remote 102
set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-id source-
packet-routing srv6 locator LOC1 end-dx2-sid 2001:db8:a1:2:101:::
set protocols bgp group IBGPv6 family evpn signaling advertise-srv6-service
set protocols bgp group IBGPv6 family evpn signaling accept-srv6-service
 routing-instances EVPN-VPWS1 route-distinguisher 65000:200
set routing-instances EVPN-VPWS1 vrf-target target:65000:200
set routing-instances EVPN-VPWS1 interface ge-0/0/1.1
```

Procedure

We describe these steps on the PE1 device. We note the differences in configurations between PE1 and PE2 when it applies. To configure EVPN-VPWS over SRv6 to support static SID, you must do the following:

1. Enable `enhanced-ip` support on all MX devices.

```
[edit]
user@PE1# set chassis network-services enhanced-ip
```

2. Configure support for SRv6 and the locator address.

   PE1

```
[edit]
user@PE1# set routing-options source-packet-routing srv6 locator LOC1 2001:db8:a1:1::/64
```

   PE2

```
[edit]
user@PE2# set routing-options source-packet-routing srv6 locator LOC1 2001:db8:a1:2::/64
```

3. Enable expanded nexthop hierarchy support for source packet routing.

```
[edit]
user@R1#set routing-options resolution preserve-nexthop-hierarchy
```

4. Enable an `evpn-vpws` routing instance.

```
[edit]
user@PE1# set routing-instances EVPN-VPWS1 instance-type evpn-vpws
```

5. Configure the SRv6 encapsulation type for the EVPN-VPWS1 routing instance.

```
[edit]
user@PE1# set routing-instances EVPN-VPWS1 protocols evpn encapsulation srv6
```

6. Configure the interface with the local and remote VPWS SID for the EVPN-VPWS1 routing instance.

   PE1

   ```
   [edit]
   user@PE1# set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-
   id local 102 remote 201
   ```

   PE2

   ```
   [edit]
   user@PE1# set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-
   id local 201 remote 102
   ```

7. Configure the locator to support END.DX2 on the interface in the EVPN-VPWS1 routing instance.

   PE1

   ```
   [edit]
   user@PE1# set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-
   id source-packet-routing srv6 locator LOC1 end-dx2-sid 2001:db8:a1:1:101::
   ```

   PE2

   ```
   [edit]
   user@PE1# set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-
   id source-packet-routing srv6 locator LOC1 end-dx2-sid 2001:db8:a1:2:101::
   ```

8. Enable the BGP protocol to advertise and to accept the EVPN NLRI for SRv6 services.

   ```
   [edit]
   user@PE1# set protocols bgp group IBGPv6 family evpn signaling advertise-srv6-service
   user@PE1# set protocols bgp group IBGPv6 family evpn signaling accept-srv6-service
   ```

9. **PE1**

Configure the vrf target and route distinguisher for the routing instance.

```
[edit]
user@PE1# set routing-instances EVPN-VPWS1 route-distinguisher 6500:100
user@PE1# set routing-instances EVPN-VPWS1 vrf-target target:65000:200
```

PE2

```
[edit]
user@PE1# set routing-instances EVPN-VPWS1 route-distinguisher 6500:100
user@PE1# set routing-instances EVPN-VPWS1 vrf-target target:65000:300
```

10. Assign the interface to the routing instance.

```
set routing-instances EVPN-VPWS1 interface ge-0/0/1.1
```

11. 

NOTE: Starting in Junos OS Release 24.1R1 and Junos OS Evolved Release 24.1R1, you do not need to configure a routing policy for EVPN-VPWS over SRv6.

Dynamic SID Allocation

Dynamic SID allocation allows you to provision the Junos device by only specifying the locator name. To enable dynamic provisioning, configure the locator name at the `[edit routing-instance` *`routing-instance-name instance-type`* `protocols evpn interface` *`interface-name`* `vpws-service-id source-packet-routing srv6]` hierarchy. The device dynamically allocates a SID with `en-dx2-sid` to the corresponding locator prefix when the service is needed. The following is the sample configuration for the dynamically allocated SID on EVPN-VPWS.

```
set routing-instances EVPN-VPWS1 instance-type evpn-vpws
set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-id local 3040
set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-id remote 20
set routing-instances EVPN-VPWS1 protocols evpn interface ge-0/0/1.1 vpws-service-id source-
packet-routing srv6 locator LOC2
set routing-instances EVPN-VPWS1 protocols evpn encapsulation srv6
set routing-instances EVPN-VPWS1 interface ge-0/0/1.1
set routing-instances EVPN-VPWS1 route-distinguisher 65000:100
set routing-instances EVPN-VPWS1 vrf-target target:65000:200
```

```
set routing-options source-packet-routing srv6 locator LOC2 2001:db8:b1:1::/64
```

RELATED DOCUMENTATION

https://www.juniper.net/documentation/en_US/day-one-books/DayOne-Intro-SRv6.pdf

Operations and Maintenance

IN THIS SECTION

Operations and Maintenance (SRv6)

Use these Operations, Administration, and Maintenance (OAM) commands to monitor Segment Routing over IPv6 (SRv6) networks and detect connectivity, forwarding, path, or policy-related issues.

- traceroute srv6—An SRv6 segment is a 128-bit value Segment Identifier (SID) often used as a shorter reference for SRv6 Segment.
- ping srv6—Check the reachability of a Segment Routing with IPv6 data plane (SRv6) network.

4

CHAPTER

Other Segment Routing Resources

IN THIS CHAPTER

Supported Standards for Segment Routing

Junos OS substantially supports the following RFCs and Internet drafts for Segment Routing.

- draft-agrawal-spring-srv6-mpls-interworking-06, *SRv6 and MPLS interworking*

  Supports service interworking.

- draft-ali-spring-sr-traffic-accounting, *SR traffic matrix accounting (Partial support)*

  Supports counter only.

- draft-bashandy-rtgwg-segment-routing-ti-lfa, *Topology Independent Fast Reroute using Segment Routing*

- draft-bashandy-rtgwg-segment-routing-uloop, *Microloop Avoidance using Segment Routing*

- draft-barth-pce-segment-routing-policy-cp-05, *PCEP extension to support Segment Routing Policy Candidate Paths*

- draft-filsfils-rtgwg-segment-routing-use-cases-02, *Segment Routing Use Cases*

- draft-filsfils-spring-sr-policy-considerations-05, *SR Policy Implementation and Deployment Considerations*

- draft-filsfils-spring-sr-traffic-counters, *SR traffic counters*

- draft-ginsberg-isis-prefix-attributes, *IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability (Partial support)*

- draft-ietf-bess-srv6-services-07, *SRv6 BGP based Overlay Services*

- draft-ietf-idr-bgp-prefix-sid, *Advertise BGP segment for a BGP Prefix*

- draft-ietf-idr-bgpls-segment-routing-epe, *BGP-LS extensions for Egress peer traffic engineering using SR*

- draft-ietf-idr-segment-routing-te-policy, *Advertise SR-TE policies via BGP*

- draft-ietf-idr-segment-routing-te-policy-09, *Advertising Segment Routing Policies in BGP*

- draft-ietf-lsr-flex-algo-11.txt, *IGP Flexible Algorithm (Partial support)*

  Supports IS-IS only.

- draft-ietf-lsr-isis-srv6-extensions, *IS-IS Extensions to Support Segment Routing over IPv6 Dataplane*

- draft-ietf-ospf-segment-routing-extensions, *OSPF extensions to distribute SR segments*

- draft-ietf-pce-segment-routing, *PCE extensions to setup a SR-TE path from the controller (south bound)*
- draft-ietf-isis-segment-routing-extensions, *ISIS extensions to distribute SR segments*
- draft-ietf-rtgwg-segment-routing-ti-lfa-04, *Topology Independent Fast Reroute using Segment Routing*
- draft-ietf-spring-conflict-resolution, *Segment Routing MPLS Conflict Resolution*
- draft-ietf-spring-ipv6-use-cases, *Use Cases for IPv6 Source Packet Routing in Networking (Partial support)*
- draft-ietf-spring-resiliency-use-cases, *Resiliency use cases in SPRING networks*
- draft-ietf-spring-segment-routing-msdc, *BGP-Prefix Segment in Large Scale data centers (Partial support)*
- draft-ietf-spring-segment-routing-central-epe, *SR Centralized BGP Egress Peer Engineering*
- draft-ietf-spring-segment-routing-mpls, *Segment Routing details with MPLS forwarding*
- draft-ietf-spring-segment-routing-policy, *SR policy for TE (Partial support)*
- draft-ietf-spring-segment-routing-policy-07.txt, *Segment Routing Policy Architecture*
- draft-kaliraj-idr-bgp-classful-transport-planes-12, *BGP Classful Transport Planes*
- RFC 4971, *Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information (Partial support)*

  Supports counter only.
- RFC 5286, *Basic Specification for IP Fast Reroute: Loop-Free Alternates*
- RFC 7471, *OSPF Traffic Engineering (TE) Metric Extensions*
- RFC 7490, *Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)*
- RFC 7684, *OSPFv2 Prefix/Link Attribute Advertisement*
- RFC 7752, *North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP*
- RFC 7855, *Source Packet Routing in Networking (SPRING) Problem Statement and Requirements*
- RFC 8102, *Remote-LFA Node Protection and Manageability*
- RFC 8277, *Using BGP to Bind MPLS Labels to Address Prefixes*

- RFC 8287, *LSP Ping/Traceroute for Segment Routing*
- RFC 8402, *Segment Routing Architecture (Partial support)*
- RFC 8403, *A Scalable and Topology-Aware MPLS Data-Plane Monitoring System*
- RFC 8426, *RSVP-SR coexistence*
- RFC 8570, *IS-IS Traffic Engineering (TE) Metric Extensions (Partial support)*

  Supports link delay related parameters.
- RFC 8571, *BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions*
- RFC 8604, *Interconnecting Millions of Endpoints with Segment Routing*
- RFC 8660, *Segment Routing with the MPLS Data Plane*
- RFC 8661, *Segment Routing MPLS Interworking with LDP*
- RFC 8663, *MPLS Segment Routing over IP (Partial support)*
- RFC 8665, *OSPF Extensions for Segment Routing*
- RFC 8690, *Clarification of Segment ID Sub-TLV Length for RFC 8287*
- draft-xu-mpls-sr-over-ip, *MPLS Segment Routing over IP (Partial support)*
- RFC 8919, *IS-IS Application-Specific Link Attributes (Partial support)*
- RFC 8986, *Segment Routing over IPv6 (SRv6) Network Programming*
- RFC 9085, *Border Gateway Protocol - Link State (BGP-LS) Extensions for Segment Routing*
- RFC 9086, *Border Gateway Protocol - Link State (BGP-LS) Extensions for Segment Routing BGP Egress Peer Engineering*
- RFC 9256, *Segment Routing Policy Architecture*
- RFC 9800 *Compressed SRv6 Segment List Encoding (Partial support).* We only support SRv6 Micro-SIDs.

The following RFCs do not define standards, but provide information about Segment Routing and related technologies. The IETF classifies them variously as “Experimental” or “Informational.”

- RFC 6571, *Loop-Free Alternate (LFA) Applicability in Service Provider (SP) Networks*
- RFC 9087, *Segment Routing Centralized BGP Egress Peer Engineering*

RELATED DOCUMENTATION

Accessing Standards Documents on the Internet