Nokia Service Router Linux 7250 Interconnect Router

Release 26.10

MPLS Guide

3HE 22255 AAAC TQZZA
Edition: 01
November 2026

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Nokia is committed to diversity and inclusion. We are continuously reviewing our customer documentation and consulting with standards bodies to ensure that terminology is inclusive and aligned with the industry. Our future customer documentation will be updated accordingly.

This document includes Nokia proprietary and confidential information, which may not be distributed or disclosed to any third parties without the prior written consent of Nokia.

This document is intended for use by Nokia's customers ("You"/"Your") in connection with a product purchased or licensed from any company within Nokia Group of Companies. Use this document as agreed. You agree to notify Nokia of any errors you may find in this document; however, should you elect to use this document for any purpose(s) for which it is not intended, You understand and warrant that any determinations You may make or actions You may take will be based upon Your independent judgment and analysis of the content of this document.

Nokia reserves the right to make changes to this document without notice. At all times, the controlling version is the one available on Nokia's site.

No part of this document may be modified.

NO WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF AVAILABILITY, ACCURACY, RELIABILITY, TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, IS MADE IN RELATION TO THE CONTENT OF THIS DOCUMENT. IN NO EVENT WILL NOKIA BE LIABLE FOR ANY DAMAGES, INCLUDING BUT NOT LIMITED TO SPECIAL, DIRECT, INDIRECT, INCIDENTAL OR CONSEQUENTIAL OR ANY LOSSES, SUCH AS BUT NOT LIMITED TO LOSS OF PROFIT, REVENUE, BUSINESS INTERRUPTION, BUSINESS OPPORTUNITY OR DATA THAT MAY ARISE FROM THE USE OF THIS DOCUMENT OR THE INFORMATION IN IT, EVEN IN THE CASE OF ERRORS IN OR OMISSIONS FROM THIS DOCUMENT OR ITS CONTENT.

Copyright and trademark: Nokia is a registered trademark of Nokia Corporation. Other product names mentioned in this document may be trademarks of their respective owners.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

© 2026 Nokia.

Table of contents

1 About this guide 6
1.1 Precautionary and information messages 6
1.2 Conventions 6

2 What's new 8

3 Overview 9
3.1 About MPLS 9
3.1.1 LSRs 9
3.2 About LDP 10
3.3 LDP IPv6 11
3.3.1 LDP operation in an IPv6 network 11
3.3.2 Link LDP 12
3.3.3 FEC resolution 12
3.4 Supported functionality 12

4 Configuring MPLS 14
4.1 MPLS label manager 14
4.1.1 Static and dynamic label blocks 14
4.1.2 Configuring label blocks 14
4.1.3 Displaying label block information 15
4.2 Static MPLS forwarding using next-hop-groups 15
4.2.1 Configuring an ingress LER 16
4.2.2 Configuring a transit LSR 16
4.2.3 Configuring PHP 17
4.3 Static MPLS forwarding using static LSP egress entries 18
4.3.1 Configuring static egress LSPs 19
4.3.2 Configuring static egress LSP traffic-class classification 21
4.4 Per-interface MPLS reception control 22
4.4.1 Enabling MPLS on a routed subinterface 22
4.5 ACLs and MPLS traffic 24
4.6 MPLS MTU 24
4.6.1 Configuring the MPLS MTU 25
4.6.2 Displaying MPLS MTU information 25

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

4.7 TTL propagation and TTL expiry 26
4.8 ICMP extensions for MPLS 26
4.8.1 ICMP extensions for transit LSRs 26
4.8.1.1 Configuring ICMP tunneling 26
4.8.2 ICMP extensions for egress LERs 27
4.9 Label values 27
4.9.1 Configuring null label for static tunnels 28
4.10 MPLS ECMP 29
4.10.1 MPLS LSR ECMP (7250 IXR platforms) 29
4.10.2 Configuring MPLS ECMP hashing options 30
4.11 Show reports for MPLS tunnel tables 30
4.12 TTM preferences for supported tunnel types 31

5 Configuring LDP 32
5.1 Enabling LDP 32
5.2 Configuring LDP neighbor discovery 32
5.3 Configuring LDP peers 33
5.4 Configuring a label block for LDP 34
5.5 Configuring implicit null label 34
5.6 Configuring longest-prefix match for IPv4 and IPv6 FEC resolution 35
5.7 Configuring load balancing over equal-cost paths 35
5.8 Configuring graceful restart helper capability 36
5.9 Configuring LDP-IGP synchronization 36
5.10 Configuring LDP tunnel-down damp time 37
5.11 Configuring LDP label withdrawal delay 38
5.12 Configuring static FEC (FEC originate) 39
5.13 Overriding the LSR ID on an interface 39
5.14 LDP FEC import and export policies 40
5.14.1 Configuring routing policies for LDP FEC import and export policies 41
5.14.2 Applying global LDP FEC import and export policies 43
5.14.3 Applying per-peer LDP FEC import and export policies 43
5.15 Targeted LDP 43
5.15.1 Configuring targeted LDP 44
5.16 Manual LDP RLFA 45
5.16.1 Configuring manual LDP RLFA 46
5.17 Automatic LDP RLFA 47

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

5.17.1 Configuring automatic LDP RLFA....49
5.17.2 Displaying source context for automatic T-LDP session parameters....50
5.17.3 Deleting redundant T-LDP sessions....51
5.18 LSP ping and trace for LDP tunnels....52

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

1 About this guide

This guide describes the services and provides configuration examples for the Multiprotocol Label Switching (MPLS) protocol used with the Nokia Service Router Linux (SR Linux).

This document is intended for users who plan to implement MPLS for SR Linux.

Note:
This manual covers the current release and may also contain some content to be released in later maintenance loads. See the *SR Linux Software Release Notes* for information about features supported in each load.

Configuration and command outputs shown in this guide are examples only; actual displays may differ depending on supported functionality and user configuration.

1.1 Precautionary and information messages

The following are information symbols used in the documentation.

DANGER: Danger warns that the described activity or situation may result in serious personal injury or death. An electric shock hazard could exist. Before you begin work on this equipment, be aware of hazards involving electrical circuitry, be familiar with networking environments, and implement accident prevention procedures.

WARNING: Warning indicates that the described activity or situation may, or will, cause equipment damage, serious performance problems, or loss of data.

Caution: Caution indicates that the described activity or situation may reduce your component or system performance.

Note: Note provides additional operational information.

Tip: Tip provides suggestions for use or best practices.

1.2 Conventions

The Nokia SR Linux documentation uses the following command conventions:

- **Bold** type indicates a command that the user must enter.
- Input and output examples are displayed in `Courier` text.
- A vertical bar (|) indicates a mutually exclusive argument.
- Square brackets ([ ]) indicate optional elements.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

- Braces ({ }) indicate a required choice. When braces are contained within square brackets, they indicate a required choice within an optional element.
- *Italic* type indicates a variable.

The following table outlines platform grouping conventions used in the SR Linux documentation suite.

Note: Some platforms in the 7250 IXR support mixed systems. For more information about mixed system support, see "Chassis types" in the *Configuration Basics Guide*.

Table 1: Platform grouping legend

Platform group	Description
7215 IXS	7215 IXS-A1[1]
7220 IXR	All 7220 IXR platforms
7220 IXR-D*x*	7220 IXR-D1, 7220 IXR-D2, 7220 IXR-D2L, 7220 IXR-D3, 7220 IXR-D3L, 7220 IXR-D4, 7220 IXR-D5
7220 IXR-H*x*	7220 IXR-H2, 7220 IXR-H3, 7220 IXR-H4, 7220 IXR-H4-32D, 7220 IXR-H5-32D, 7220 IXR-H5-64D, 7220 IXR-H5-64O, 7220 IXR-H5-64O-512, 7220 IXR-H6-64
7250 IXR[2]	7250 IXR platforms
7250 IXR Gen 2	7250 IXR-6, 7250 IXR-10
7250 IXR Gen 2c+	7250 IXR-6e with IMM2, 7250 IXR-10e with IMM2, 7250 IXR-X1b, 7250 IXR-X3b
7250 IXR Gen 3	7250 IXR-6e with IMM3, 7250 IXR-10e with IMM3, 7250 IXR-18e, 7250 IXR-X4, 7250 IXR-X4 OSFP
7250 IXR-6e/10e (mixed system)	7250 IXR-6e (mixed system)[3], 7250 IXR-10e (mixed system)[3]

1 References to 7215 IXS may be appended with (AA variant) or (AB variant) to indicate the chassis variant. Both variants are 7215 IXS platforms; they are identified by part numbers ending with AA or AB.

2 References to the 7250 IXR platform group may be appended with (including mixed systems) or (excluding mixed systems) to indicate support for a mix of 7250 IXR Gen 2c+ (IMM2) and 7250 IXR Gen 3 (IMM3) line cards in the same chassis.

3 References to this platform as part of 7250 IXR (mixed system) indicate mixed system support of 7250 IXR Gen 2c+ (IMM2) and 7250 IXR Gen 3 (IMM3). That is, the 7250 IXR-6e and 7250 IXR-10e can hold and support both IMM2 and IMM3 at the same time.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

2 What's new

This section lists the changes that were made in this release.

Table 2: What's new in Release 26.10.1

Topic	Location
Static MPLS forwarding using static LSP egress entries as an alternative to next-hop-groups.	Static MPLS forwarding using static LSP egress entries
Enable/disable MPLS on specific interfaces	Supported functionality

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

3 Overview

The following provides a brief description of MPLS and LDP and lists the functionality supported by SR Linux in this release.

3.1 About MPLS

Note: MPLS is supported on 7250 IXR platforms:

Multiprotocol Label Switching (MPLS) is a label switching technology that provides the ability to set up connection-oriented paths over a connectionless IP network. MPLS facilitates network traffic flow and provides a mechanism to engineer network traffic patterns independently from routing tables. MPLS sets up a specific path for a sequence of packets. The packets are identified by a label stack inserted into each packet.

MPLS requires a set of procedures to enhance network-layer packets with label stacks, which then turns them into labeled packets. Routers that support MPLS are known as Label Switching Routers (LSRs). To transmit a labeled packet on a particular data link, an LSR must support the encoding technique.

In MPLS, packets can carry not only one label but a set of labels in a stack. An LSR can swap the label at the top of the stack, pop the label, or swap the label and push one or more labels into the stack. The processing of a labeled packet is completely independent of the level of hierarchy. The processing is always based on the top label, without regard for the possibility that some number of other labels may have been above it in the past, or that some number of other labels may be below it at present.

3.1.1 LSRs

Label switch routers (LSRs) perform different label switching functions based on their position in a Label Switched Path (LSP). The LSRs in an LSP do one of the following:

- The LSR at the or head-end of an LSP is the ingress label edge router (LER). The ingress LER can encapsulate packets with an MPLS header and forward them to the next router along the path. A point-to-point LSP can only have one ingress LER.

  The ingress LER is programmed to perform a label push operation. More specifically, it is programmed with an IP route that encapsulates matching IP packets using MPLS and forwards them to one or more next-hop routers. Each outgoing MPLS packet has a label stack (in the MPLS header), and the top entry in each stack contains a label value pushed by the route lookup process that indicates the path to be followed.

- A transit LSR is an intermediate router in the LSP between the ingress and egress routers. Each transit LSR along the path is programmed to perform a label swap operation: the transit LSR is programmed with an MPLS route that matches the label value at the top of the label stack, pops that label stack entry, pushes a new top label and forwards the resulting MPLS packet to the next set of routers along the path.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

- The penultimate LSR (the one before last) in the LSP can be programmed to perform a pop-and-forward operation, known as penultimate hop popping (PHP). In this case, the LSR is programmed with an MPLS route that matches the label value at the top of the label stack, pops that label stack entry, and forwards the resulting packet to the tail-end router. The packet sent by the PHP LSR to the egress LER may be the original IP payload packet, but the forwarding decision made by the PHP LSR is based only on the incoming top label value, not IP route lookup.
- The router at the tail-end of an LSP is the egress LER. The egress LER strips the MPLS encapsulation, which changes it from an MPLS packet to a data packet, and then forwards the packet to its destination using information in the forwarding table. Each point-to-point LSP can have only one egress router. The ingress and egress routers in an LSP cannot be the same router.

  If the penultimate LSR is just a normal transit LSR performing a label swap (no PHP), the egress LER must be programmed to perform the pop operation. In this case, the programmed MPLS route matches the label value at the top of the label stack, the egress LER pops that label entry and then does another lookup on the next header; usually this is an IP header and the packet is forwarded based on IP route lookup.

3.2 About LDP

Label Distribution Protocol (LDP) is a protocol used to distribute MPLS labels in non-traffic-engineered applications. LDP allows routers to establish LSPs through a network by mapping network-layer routing information directly to data link layer-switched paths.

An LSP is defined by the set of labels from the ingress LER to the egress LER. LDP associates a Forwarding Equivalence Class (FEC) with each LSP it establishes. A FEC is a collection of common actions associated with a class of packets. When an LSR assigns a label to a FEC, it must allow other LSRs in the path to know about the label. LDP helps to establish the LSP by providing a set of procedures that LSRs can use to distribute labels.

The FEC associated with an LSP specifies which packets are mapped to that LSP. LSPs are extended through a network as each LSR splices incoming labels for a FEC to the outgoing label assigned to the next-hop for the specific FEC.

LDP performs the label distribution only in MPLS environments. The LDP operation begins with a hello discovery process to find LDP peers in the network. LDP peers are two LSRs that use LDP to exchange label/FEC mapping information. An LDP session is created between LDP peers. A single LDP session allows each peer to learn the other’s label mappings (LDP is bidirectional) and to exchange label binding information.

LDP signaling works with the MPLS label manager to manage the relationships between labels and the corresponding FEC. An MPLS label identifies a set of actions that the forwarding plane performs on an incoming packet before discarding it. The FEC is identified through the signaling protocol (in this case, LDP) and allocated a label. The mapping between the label and the FEC is communicated to the forwarding plane.

When an unlabeled packet ingresses the router, classification policies associate it with a FEC. The appropriate label is imposed on the packet, and the packet is forwarded.

Supported platforms

Link LDP is supported on 7250 IXR platforms.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Table 3: LDP platform support

Functionality	7250 IXR Gen 2c+	7250 IXR Gen 3
l-LDP v4, IS-IS	supported	supported
l-LDP v6, IS-IS	supported	supported
l-LDP v4, OSPF	supported	supported
l-LDP v6, OSPFv3	supported	supported

3.3 LDP IPv6

SR Linux extends the LDP control plane and data plane to support LDP IPv6 adjacencies and sessions using 128-bit LSR IDs.

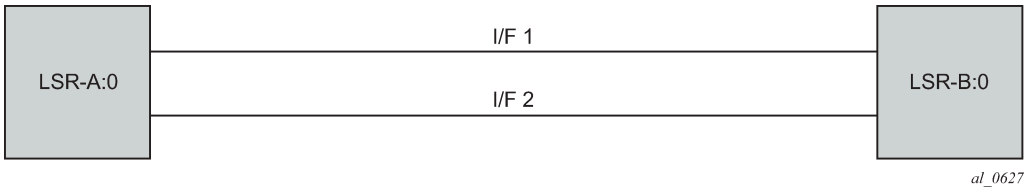
LDP IPv6 is supported for both link and targeted LDP. On 7250 IXR, link LDP IPv6 is not supported. Targeted LDP with IPv6 NHs is supported with IS-IS.

The implementation allows for concurrent support of independent LDP IPv4 (32-bit LSR ID) and IPv6 (128-bit LSR ID) adjacencies and sessions between peer LSRs.

3.3.1 LDP operation in an IPv6 network

LDP IPv6 can be enabled on an SR Linux subinterface. The following figure shows the LDP adjacency and session over an IPv6 subinterface.

Figure 1: LDP adjacency and session over an IPv6 subinterface



LSR-A and LSR-B have the following IPv6 LDP identifiers respectively:

- <LSR ID=A/128> : <label space id=0>
- <LSR ID=B/128> : <label space id=0>

By default, A/128 and B/128 use the system subinterface IPv6 address.

Note: Although the LDP control plane can operate using only the IPv6 system address, you must configure the IPv4-formatted router ID for IS-IS to operate properly.

The following sections describe the behavior when LDP IPv6 is enabled on the subinterface.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

3.3.2 Link LDP

The SR Linux LDP IPv6 implementation uses a 128-bit LSR ID as defined in RFC 5036.

The Hello adjacency is brought up using link Hello packets with a source IP address set to the subinterface link-local unicast address and a destination IP address set to the link-local multicast address FF02:0:0:0:0:0:0:2.

The transport address for the TCP connection, which is encoded in the Hello packet, is set to the LSR ID of the LSR by default.

3.3.3 FEC resolution

LDP advertises and withdraws all subinterface IPv6 addresses using the Address/Address-Withdraw message. Both the link-local unicast address and the configured global unicast addresses of an subinterface are advertised.

The LSR does not advertise a FEC for a link-local address and, if received, the LSR does not resolve it.

An IPv4 or IPv6 prefix FEC can be resolved to an LDP IPv6 subinterface in the same way it is resolved to an LDP IPv4 subinterface. The outgoing subinterface and next hop are looked up in the RTM cache. The next hop can be the link-local unicast address of the other side of the link or a global unicast address. The FEC is resolved to the LDP IPv6 subinterface of the downstream LDP IPv6 LSR that advertised the IPv4 or IPv6 address of the next hop.

3.4 Supported functionality

In 26.0.1 release, SR Linux supports the following MPLS and LDP functionalities:

MPLS

- Statically configured MPLS forwarding entries
- Configurable label range
- MPLS label manager that shares the MPLS label space among client applications that require MPLS labels
- Per-interface enabling and disabling of ingress MPLS packet reception on routed subinterfaces

LDP

- LDPv4 implementation compliant with RFC 5036
- LDPv6 implementation compliant with RFC 5036 and 7552
- LDP in the default network-instance only
- Label distribution using DU (downstream unsolicited), ordered control
- Platform label space only
- Configurable label range (dynamic and non-shared label block)
- Overload TLV when label-block has no free entries
- Configurable timers (hello-interval, hello-holdtime, and keepalive-interval)

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

- Ingress LER, transit LSR, and egress LER roles for /32 IPv4 or /128 IPv6 FECs
- Automatic FEC origination of the system address
- /32 IPv4 FEC resolution using IGP routes, with the longest prefix match option
- /128 IPv6 FEC resolution using IGP routes, with the longest prefix match option
- ECMP with configurable next-hop limit (up to 64)
- Automatic installation of all LDP /32 IPv4 and /128 IPv6 prefix FECs to TTM
- Per-peer configurable FEC limit
- Graceful restart helper capability
- BGP shortcuts over LDP tunnels
- Protocol debug/trace-options
- LDP-IGP synchronization
- Advertise Address Mapping message for primary IPv4 or IPv6 address of the adjacent subinterface only
- Non-configurable capability advertisement in Initialization messages that support the following:
  - State Advertisement Control (SAC) for the IPv4 prefix FECs only
  - Fault tolerance (Graceful Restart)
  - Nokia-overload TLV
  - Unrecognized notification
- Split-horizon : A label-mapping message is not advertised to a peer if the FEC matches an address sent by that peer in an Address Mapping message.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

4 Configuring MPLS

This chapter provides information about how MPLS functions on SR Linux and examples of common configuration tasks.

4.1 MPLS label manager

SR Linux features an MPLS label manager process that shares the MPLS label space among client applications that require MPLS labels; these applications include static MPLS forwarding and LDP.

For a protocol such as LDP to become operational, it must be configured with a reference to a predefined range of labels, called a label block. A label block configuration includes a start-label value and an end-label value. When the label block is made available to an application, the application can use any label in the range between the start-label and end-label, inclusive of those values.

The MPLS label manager ensures there is no configuration overlap between the labels of two different label blocks.

4.1.1 Static and dynamic label blocks

A label block can be static or dynamic:

- A static label block is provided by the MPLS label manager to a client application when it is expected that the client application specifies the exact label value it wants to use with every label request.
- A dynamic label block is provided by the MPLS label manager to a client application when it is expected that the client application requests the next available label when it needs a new entry.

A label block can be configured as shared or dedicated. When a label block is configured to be shared, it allows the label block to be made available to multiple protocols. If a label block is not configured as shared, it is reserved for the exclusive use of one protocol.

The label block used by LDP must be configured as a dynamic, non-shared label block. For static MPLS routes, it is necessary to configure a static label block and reference the static label block in the static MPLS configuration.

4.1.2 Configuring label blocks

Procedure

To configure a static or dynamic label block, you specify the start and end label for the range.

Example

The following example configures a static and dynamic label block.

```
--{ + candidate shared default }--[  ]--
# info with-context system mpls
    system {
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
        mpls {
            label-ranges {
                static s1 {
                    start-label 10001
                    end-label 11000
                }
                dynamic d1 {
                    start-label 11001
                    end-label 12000
                }
            }
        }
    }
```

4.1.3 Displaying label block information

Procedure

Use the **info from state** command to display information about the configured label blocks.

Example

```
--{ + candidate shared default }--[  ]--
# info from state with-context system mpls label-ranges
    system {
        mpls {
            label-ranges {
                static s1 {
                    shared true
                    start-label 10001
                    end-label 11000
                    allocated-labels 0
                    free-labels 1000
                    status ready
                }
                dynamic d1 {
                    start-label 11001
                    end-label 12000
                    allocated-labels 0
                    free-labels 1000
                    status ready
                }
            }
        }
    }
```

4.2 Static MPLS forwarding using next-hop-groups

Statically configured MPLS forwarding entries allow MPLS-labeled packets to be forwarded along a specific path that may differ from the normal shortest path provided by the underlay routing protocol.

Static next-hop-groups used by IPv4 and IPv6 static routes of the default network-instance support MPLS next hops. An MPLS next-hop has a next-hop IP address and a list of MPLS labels (currently limited to 1). When an IP packet matches a static route pointing to a next-hop-group with MPLS next-hops, the packet is MPLS encapsulated according to the selected next-hop.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Static MPLS routes are supported in the default network-instance. Static MPLS routes control the way that transit LSRs, penultimate LSRs, and egress LERs handle received MPLS packets. Each static MPLS route matches a particular label value and causes that label value to be popped from the label stack when it appears as the top label in any received MPLS packet, on any interface. If the static MPLS route points to a next-hop-group with MPLS next-hops, the packet is forwarded to the selected next-hop with a swap operation; ECMP is supported if there are multiple MPLS next-hops. When the **pushed-mpls-label-stack** parameter for the MPLS next-hop specifies the IPv4 or IPv6 `IMPLICIT_NULL` label value, no new label is pushed and a PHP pop-and-forward operation is performed.

4.2.1 Configuring an ingress LER

Procedure

To configure an ingress LER, you specify the label to push to MPLS-encapsulated packets.

Example

The following example shows the default network-instance configured for an ingress LER, specifying the label to push to MPLS-encapsulated packets. In this example, multiple next-hops (ECMP) are specified in a next-hop group.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default next-hop-groups group srva_tora_ipv4
    network-instance default {
        next-hop-groups {
            group srva_tora_ipv4 {
                nexthop 0 {
                    ip-address 192.13.1.3
                    resolve false
                    pushed-mpls-label-stack [
                        544334
                    ]
                }
                nexthop 2 {
                    ip-address 192.13.1.3
                    resolve false
                    pushed-mpls-label-stack [
                        205679
                    ]
                }
            }
        }
    }
```

4.2.2 Configuring a transit LSR

Procedure

To configure a transit LSR with an explicit swap operation, you specify a label block for MPLS and a next-hop to use for label-swap operations.

Example: Specify a static label block for MPLS

The following example shows the default network-instance configured as a transit LSR in the label switched path.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

This example configures MPLS to use a static label block named `s2`. The static label block is configured with a start-label value and an end-label value. See Configuring label blocks for an example of a static label block configuration.

The example configures incoming MPLS transit traffic with label 1000 to use the next-hops in next-hop-group `nhop_group_1` for label-swap operations.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default mpls
    network-instance default {
        mpls {
            static-label-block s2
            static-entry 1000 preference 100 {
                operation swap
                next-hop-group nhop_group_1
            }
        }
    }
```

Example: Specify next-hop for MPLS

The following example specifies the MPLS next-hop for `nhop_group_1`. Only one MPLS next-hop is supported per next-hop-group. In this example, the label for outgoing traffic to MPLS next-hop 192.35.1.5 is swapped to 1001.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default next-hop-groups group nhop_group_1
    network-instance default {
        next-hop-groups {
            group nhop_group_1 {
                nexthop 0 {
                    ip-address 192.35.1.5
                    resolve false
                    pushed-mpls-label-stack [
                        1001
                    ]
                }
            }
        }
    }
```

4.2.3 Configuring PHP

Procedure

To configure PHP, you configure MPLS to pop the label for outgoing traffic for a next-hop.

Example

The following example shows the default network-instance configured to perform PHP for the LSR. In this example, the setting for `pushed-mpls-label-stack` is `IMPLICIT_NULL`, which causes the label for outgoing traffic to MPLS next-hop 192.35.1.1 to be popped.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default next-hop-groups group nhop_group_2
    network-instance default {
        next-hop-groups {
            group nhop_group_2 {
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
                nexthop 0 {
                    ip-address 192.35.1.1
                    resolve false
                    pushed-mpls-label-stack [
                        IMPLICIT_NULL
                    ]
                }
            }
        }
    }
```

4.3 Static MPLS forwarding using static LSP egress entries

Note: Static LSP entries are supported only on 7250 IXR Gen 3 platforms.

Note: Nokia recommends to configure only one static MPLS forwarding mechanism (next-hop groups or static LSP egress entries) at the same time.

To configure static MPLS forwarding, you can configure static LSP entries associating an incoming label with an IPv4 or IPv6 next-hop address and network instance. The static incoming label entries are supported only on egress LERs, defining a pop-and-forward operation.

- The static LSP creates a pop-and-forward ILM entry: the top label stack entry is removed and the packet is forwarded out the selected next hop with its remaining MPLS label stack entries (if any), its IP headers, and its payload.
- The outgoing packet can be an IPv4 unicast, IPv6 unicast, or IPv4 multicast packet with one or more labels.
- The static LSP is created in the default network instance and the outgoing interface is an IP interface that must be in a different (non-default) IP-VRF network instance. The static LSP configuration must reference the IP-VRF network instance containing the outgoing IP interface. Because different IP VRFs can have overlapping IP addresses, specifying an IP address on its own is insufficient to determine the egress IP interface.
- The next-hop IP address of the static LSP can specify only a direct next hop.

Route resolution

Together, the configured **ip-address** and **network-instance** identify a single outgoing interface for the direct next hop. Routes are resolved as follows:

- No restrictions exist on the type of routes used to resolve the next-hop IP address.
- Route resolution is limited to one level. If the route that resolves the next-hop IP address has an indirect next hop, the next hop is considered down.
- Full ECMP route resolution is not supported. If the route that resolves the next-hop IP address is an ECMP route with multiple direct next hops, SR Linux deterministically selects just one of those next hops to program the ILM entry.
- If all the next hops are unresolved, a static LSP becomes inactive and is deprogrammed from the MPLS ILM table. This is visible using the **info from state** command, which displays the LSP **oper-state** as **down** and the **oper-down-reason** as **next-hops-unresolved**.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Label allocation

Labels for static LSP entries are allocated from a dedicated static label block, specified using the **network-instance mpls static-label-block** command. If the static label block is not configured, static LSP incoming labels are allocated from the default label block (default range of 16 to 1048575).

- The default label block is a non-shared static label range automatically created by the label manager if the system has no user-configured dynamic or static label block configured under **system mpls label-ranges**. The default label block is not visible in the running configuration or in state.
- If a dynamic or static label block is present under **system mpls label-ranges**, you must specify a static label range for the static LSP entries to use under **network-instance mpls static-label-block**; otherwise, the configuration of static LSPs is blocked.
- SR Linux supports the configuration of multiple static LSPs with the same **incoming-label**, but only one ILM for that label is installed. Assuming all next hops are resolvable, static LSPs with identical **incoming-label** get sorted by name (in ascending order) and only the first one is installed. The other next hops display **oper-down-reason label-unavailable** in **info from state**.
- If the MPLS incoming-labels resource is exhausted, static LSPs that cannot be programmed as ILM entries are inactive. In this case, the **oper-state** of the LSP in **info from state** is shown as **down** with an **oper-down-reason** of **resource-allocation-failed**.

MPLS packet datapath behavior

After popping the top label, if the exposed packet is still an MPLS packet with one or more labels:

- The MPLS TTL in the new top label is decremented by 1.
- The destination MAC address of the transmitted packet is found from the ARP/ND entry for the static LSP next hop.

IPv4/IPv6 unicast or IPv4 multicast datapath behavior

After popping the top label, if the exposed packet is an IPv4/IPv6 unicast or IPv4 multicast packet:

- The MPLS TTL is not copied to the IP TTL (no TTL propagation).
- The IPv4 DSCP or IPv6 traffic class is not modified.
- The IP TTL is decremented by one when the packet is forwarded.
- When the **payload-type** is set to **unicast**, the destination MAC address of the transmitted IPv4/IPv6 packet is found from the ARP/ND entry for the static LSP next hop.
- When the **payload-type** is set to **multicast**, the destination MAC address of the transmitted IPv4/IPv6 packet is derived from the IP multicast address in the IP header destination address field; for example, if the destination address carries the group address 239.1.1.1, the destination MAC address is 01-00-5E-01-01-01.

4.3.1 Configuring static egress LSPs

Procedure

To configure static egress LSPs, use the **network-instance default mpls lsps static-lsp** command. The **incoming-label** must specify a valid MPLS label from within the **network-instance mpls static-label-block** or the default label block.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

If the incoming label of a static LSP is already in use by another static LSP, the new LSP status is inactive and shown in **info from state** as **oper-state down** with an **oper-down-reason** of **label-unavailable**.

Example: Define label block for static LSP entries

The following example specifies a static label block to use for the static LSPs. The **mpls static-label-block** command references a static label block configured under **system mpls label-ranges**.

```
--{ candidate shared default }--[  ]--
# info with-context network-instance default mpls static-label-block
    network-instance default {
        mpls {
            static-label-block static-egress-lsp-block-1
        }
    }
```

Example: Configure static LSP entries

The following example configures a static LSP entry for unicast traffic.

```
--{ candidate shared default }--[  ]--
# info with-context network-instance default mpls lsps
    network-instance default {
        mpls {
            lsps {
                static-lsp static-lsp-1 {
                    payload-type unicast
                    egress {
                        incoming-label 17001
                        network-instance ip-vrf-1
                        pipe-mode true
                        next-hop 10 {
                            ip-address 10.20.20.1
                        }
                    }
                }
            }
        }
    }
```

The static egress LSP parameters are as follows:

- **payload-type** — specifies the payload type: unicast (default) or multicast.
- **egress incoming-label** — specifies a valid and available MPLS label value from the configured static label block or the default label block.
- **egress network-instance** — specifies a locally configured IP-VRF network instance. Each **next-hop** IP address is resolved to a routed subinterface in the specified network instance. Because different IP VRFs can have overlapping IP addresses, **ip-address** on its own is insufficient to determine the egress IP interface.
- **egress pipe-mode** — controls the QoS classification and marking of packets matching the MPLS ILM entry. When set to **true**, packets matching the static LSP ILM entry are classified and marked based on the traffic-class/EXP value of the label-stack entry that is popped. No copying of these fields to the IP header is performed. A setting of **false** (implying uniform mode) is not supported.
- **egress next-hop** *index* — specifies an index value for the next hop. A maximum of one next hop is supported.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

- **egress next-hop ip-address** — specifies an IPv4 or IPv6 address. IPv4 addresses can be global or link-local, while IPv6 addresses can be global-unicast or a link-local address with zone (for example: fe80::1234%ethernet-1/1.1). A pop-and-forward operation is performed to the specified IP address. That is, the top label stack entry is removed and the packet is forwarded out the selected next hop with its remaining MPLS label stack entries (if any), its IP headers, and its payload.

4.3.2 Configuring static egress LSP traffic-class classification

Procedure

To classify traffic entering the static egress LSP, you can use the **qos classifiers static-mpls-lsp-egress** command to bind an MPLS traffic-class classification policy (defined by the **mpls-traffic-class-policy** command) to all static egress LSPs.

The **static-mpls-lsp-egress** classification policy affects all ingress MPLS packets that are matched by an MPLS ILM entry with either of the following two operations:

- pop-xconnect-unicast
- pop-xconnect-multicast

Every static MPLS LSP configured under the **network-instance mpls lsps static-lsp egress** context creates one of these two types of ILM entries. No other method of creating MPLS ILMs (configuration or signaling protocol) can create these types of ILMs.

If an MPLS packet that matches a static egress LSP entry arrives on a subinterface that has an **mpls-traffic-class-policy** applied directly (through configuration of **qos interfaces interface input classifiers mpls-traffic-class-policy**), the packet classification is based on the **static-mpls-lsp-egress policy**, if one exists. If no **static-mpls-lsp-egress** policy is defined, the system does not fall back to the MPLS traffic-class policy defined for the subinterface. Instead, the packets are classified by the default MPLS traffic-class classifier policy, which maps each MPLS TC value one-to-one to the forwarding class of the same index: TC 0 to fc0, TC 1 to fc1, and so on through TC 7 to fc7. A drop probability of low is applied to all classes.

Example: Configure static egress LSP traffic-class classification

The following example configures an MPLS traffic class policy mapping a traffic class to a specific forwarding class and drop probability and binds that policy to all static egress LSPs.

```
--{ candidate shared default }--[  ]--
# info with-context qos classifiers static-mpls-lsp-egress MPLS_POLICY
qos {
    classifiers {
        static-mpls-lsp-egress MPLS_POLICY
        mpls-traffic-class-policy MPLS_POLICY {
            traffic-class 2 {
                forwarding-class AF3
                drop-probability low
            }
        }
    }
}
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

4.4 Per-interface MPLS reception control

Note: Per-interface MPLS reception control is supported only on 7250 IXR Gen 3 platforms.

Per-interface MPLS reception control allows a node to accept and forward ingress MPLS packets on core-facing routed subinterfaces while blocking MPLS packet reception on customer-facing routed subinterfaces. This feature is supported in the **default** network-instance and in **ip-vrf** network-instances using the **network-instance interface mpls forward-received-packets** command.

The setting applies only to ingress MPLS packets received on the routed subinterface. It does not affect the transmission of egress MPLS packets from the subinterface.

In a **default** network-instance, the implicit default value of the **network-instance interface mpls forward-received-packets** is `true`. In a network-instance of type **ip-vrf**, the implicit default value is `false`, and `true` is not supported.

In the **default** network-instance, you can control MPLS packet reception at the network-instance level using the **network-instance mpls-forwarding forward-received-packets** command.

When you configure per-interface reception control for a subinterface, the network-instance level setting is overridden for that subinterface, as follows:

- If the per-interface reception control **network-instance interface mpls forward-received-packets** is set, that value applies and overrides the network-instance setting for that subinterface.
- If it is not set, the subinterface uses the **network-instance mpls-forwarding forward-received-packets** setting at the network-instance level.

When MPLS is enabled on a routed subinterface, Ethernet-MPLS packets received on that subinterface are forwarded according to Incoming Label Map (ILM) lookup. When MPLS is disabled on a routed subinterface, Ethernet-MPLS packets received on that subinterface are discarded and counted in both **interface subinterface statistics in-discarded-packets** and **interface subinterface mpls statistics in-discarded-packets**.

The **network-instance interface mpls forward-received-packets** setting does not prevent configuring a static LSP with a next-hop on a subinterface for which MPLS reception is disabled. Also, it does not prevent a subinterface with MPLS reception disabled from being used as an LDP, RSVP-TE, or SR-MPLS interface. The setting does not affect control-plane signaling protocols.

For configuration examples, see Enabling MPLS on a routed subinterface.

4.4.1 Enabling MPLS on a routed subinterface

Procedure

You can enable or disable ingress MPLS packet reception on a routed subinterface in a network-instance of type `default` or `ip-vrf` using the **network-instance interface mpls forward-received-packets** command. For more information, see Per-interface MPLS reception control.

In the `default` network-instance, the implicit default for `network-instance interface mpls forward-received-packets` is `true`, so ingress MPLS packets are accepted and forwarded on routed subinterfaces.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

To disable MPLS packet reception on a customer-facing subinterface without changing other subinterfaces, set **forward-received-packets** to `false` for that subinterface under the **network-instance interface mpls** context.

Example: Disable MPLS reception on a customer-facing subinterface

The following example disables ingress MPLS packet reception on subinterface `ethernet-1/1.1` in the `default` network-instance.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default interface ethernet-1/1.1 mpls
    network-instance default {
        interface ethernet-1/1.1 {
            mpls {
                forward-received-packets false
            }
        }
    }
```

Example: Enable MPLS reception on a routed subinterface

The following example enables ingress MPLS packet reception on subinterface `ethernet-1/2.1` in the `default` network-instance.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default interface ethernet-1/2.1 mpls
    network-instance default {
        interface ethernet-1/2.1 {
            mpls {
                forward-received-packets true
            }
        }
    }
```

Example: Display MPLS discard counters for a subinterface

When MPLS reception is disabled on a routed subinterface, Ethernet-MPLS packets received on that subinterface are dropped. Use the **info from state with-context** command to display the subinterface and the MPLS discard counters.

```
--{ + candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/1 subinterface 1 statistics in-
discarded-packets
    interface ethernet-1/1 {
        subinterface 1 {
            statistics {
                in-discarded-packets 12
            }
        }
    }
--{ + candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/1 subinterface 1 mpls statistics in-
discarded-packets
    interface ethernet-1/1 {
        subinterface 1 {
            mpls {
                statistics {
                    in-discarded-packets 0
                }
            }
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
        }
    }
```

4.5 ACLs and MPLS traffic

Table 4: How MPLS traffic is handled by SR Linux ACLs lists how traffic along an MPLS datapath is evaluated by each type of ACL that can be configured on SR Linux. See the *SR Linux ACL and Traffic Steering Guide* for information about configuring ACLs on SR Linux.

Table 4: How MPLS traffic is handled by SR Linux ACLs

MPLS datapath	Evaluated by ingress ACL rules?	Evaluated by egress ACL rules?	Evaluated by CPU QoS entries?	Evaluated by IP CPM filter rules?	Evaluated by IP capture filter rules?
IP → MPLS (LER)	Yes	No	N/A	N/A	Yes
MPLS → MPLS (LSR)	Yes	No	N/A	N/A	No
MPLS → term (label TTL expiry)	Yes	N/A	No	No	No
MPLS → IP (PHP)	Yes	Yes	N/A	N/A	No
MPLS → IP (UHP)	Yes	Yes	N/A	N/A	Yes
MPLS → IP → term (IP address is local)	Yes	N/A	Yes	Yes	Yes

4.6 MPLS MTU

The MPLS MTU defines the maximum-sized MPLS packet that can be transmitted from a routed subinterface, including the size of the transmitted label stack (4 bytes * number of label stack entries). If an MPLS packet containing any payload exceeds this size, the packet is dropped.

SR Linux supports a system-wide default MPLS MTU value. If you configure a default MPLS MTU value, that value is programmed as the operational MPLS MTU on all IP interfaces of the system. The supported range for the default MPLS MTU is 1284 to 9496 bytes; default 1508 bytes.

In addition to the system-wide default MPLS MTU, you can configure a subinterface-level MPLS MTU, which applies to subinterfaces of type routed. The supported range for the subinterface-level MPLS MTU is 1284-9496 bytes. If no MPLS MTU is configured for a subinterface, the default MTU value is taken from the system-wide default MPLS MTU.

Each 7250 IXR IMM supports a maximum of four different MPLS MTU values. If a line card already has four different MPLS MTU values, and a fifth MPLS MTU value is configured for a subinterface on the same line card, the subinterface is brought down with the reason `mpls-mtu-resource-exceeded`.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

4.6.1 Configuring the MPLS MTU

Procedure

You can configure a system-wide default MPLS MTU value, and you can configure a subinterface-level MPLS MTU that applies to subinterfaces of type routed. The supported range for the MPLS MTU is 1284 to 9496 bytes; default 1508 bytes. If no MPLS MTU is configured for a subinterface, the default MPLS MTU value is taken from the system-wide MPLS MTU.

Example: Configure a system-wide default MPLS MTU value

```
--{ + candidate shared default }--[  ]--
# info with-context system mtu
    system {
        mtu {
            default-mpls-mtu 4096
        }
    }
```

Example: Configure an MPLS MTU for a routed subinterface

```
--{ + candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 subinterface 1
    interface ethernet-1/1 {
        subinterface 1 {
            type routed
            admin-state enable
            mpls-mtu 4096
            ipv4 {
                admin-state enable
                address 192.168.11.1/30 {
                }
            }
            ipv6 {
                admin-state enable
                address 2001:1::192:168:11:1/126 {
                }
            }
        }
    }
```

4.6.2 Displaying MPLS MTU information

Procedure

Use the **info from state** command to display MPLS MTU information.

Example: Display the system-wide default MPLS MTU value

```
--{ + candidate shared default }--[  ]--
# info from state with-context system mtu default-mpls-mtu
    system {
        mtu {
            default-mpls-mtu 4096
        }
    }
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Example: Display the MPLS MTU for a subinterface

```
--{ + candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/1 subinterface 1 mpls-mtu
    interface ethernet-1/1 {
        subinterface 1 {
            mpls-mtu 4096
        }
    }
```

4.7 TTL propagation and TTL expiry

SR Linux supports the Uniform model for TTL propagation as described RFC 3032 and RFC 3443. The Uniform model makes all the nodes that an LSP traverses visible to nodes outside the tunnel.

4.8 ICMP extensions for MPLS

Note: This feature is supported on 7250 IXR devices.

SR Linux MPLS support includes ICMP extensions for transit LSRs and egress LERs.

- ICMP extensions for transit LSRs
- ICMP extensions for egress LERs

The ICMP extensions, defined in RFC 4950, are by default always enabled.

4.8.1 ICMP extensions for transit LSRs

When a transit LSR receives an MPLS packet that cannot be forwarded (for example, the label TTL has expired or the egress subinterface MPLS MTU was exceeded), the packet is extracted to the CPM, which attempts to find an IP packet under the remaining label stack. If an IP packet is found, the CPM generates the appropriate error message, such as time-exceeded, destination-unreachable, or parameter-problem.

For time-exceeded and destination-unreachable messages only, the CPM generates a multipath ICMPv4/ICMPv6 time-exceeded message with the label stack object of RFC 4950. This object encodes the entire MPLS label stack as it was received by the LSR that sends the ICMP message and at least 128 bytes of the original datagram (zero padded to this length if necessary).

4.8.1.1 Configuring ICMP tunneling

Procedure

ICMP tunneling is disabled by default. You can enable it for transit LSRs; on egress LERs, the setting for the **icmp-tunneling** option is not relevant.

If the **icmp-tunneling** option is disabled, all ICMP messages (including but not limited to multipath ICMP messages constructed according to RFC 4950) are injected into the default network instance for forwarding back to the source. The attempt to find a route to the source may be unsuccessful however.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

If the **icmp-tunneling** option is enabled, all ICMP messages (including but not limited to multipath ICMP messages constructed according to RFC 4950) are injected in the forward direction of the LSP (that is, by re-adding the received label stack, resetting the MPLS TTL in all label stack entries to 255, and performing an ILM lookup on the top label). The source address of the ICMP message is an IP address of the LSR.

Example

The following example enables ICMP tunneling for a transit LSR.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default
    network-instance default {
        mpls {
            icmp-tunneling true
        }
    }
```

4.8.2 ICMP extensions for egress LERs

If the egress LER receives an MPLS packet that cannot be forwarded (for example, the label TTL has expired, the IP TTL has expired, or the egress subinterface IP MTU was exceeded) the packet is extracted to the CPM with an indication of the network-instance associated with the last popped label. The CPM generates the appropriate ICMP error message, such as time-exceeded, destination-unreachable, and so on.

For time-exceeded and destination-unreachable messages only, the CPM generates a multipath ICMPv4/ICMPv6 time-exceeded message with the label stack object of RFC4950. This object encodes the entire MPLS label stack as it was received by the egress LER and at least 128 bytes of the original datagram (zero padded to this length if necessary).

All ICMP messages (including but not limited to multipath ICMP messages constructed according to RFC 4950) are injected into the network instance associated with the last popped label for forwarding back to the source.

If the egress LER receives an MPLS packet, pops the remaining label stack, and finds an IP packet that must be forwarded to a host address on a local subnet, the packet may be queued if there is no ARP entry for the host address. If ARP cannot learn the MAC address of the host after about 3 seconds, the egress LER sends a destination-unreachable/host-unreachable message back to the source.

4.9 Label values

A packet traveling along an LSP is identified by its label, the 20-bit unsigned integer. The range is 0 to 1 048 575. Label values 0 to 31 are reserved and are defined as follows:

- A value of 0 represents the IPv4 explicit null label. It indicates that the label stack must be popped, and the packet forwarding must be based on the IPv4 header.
- A value of 1 represents the router alert label. The label value is legal anywhere in the label stack except at the bottom. When a received packet contains this label value at the top of the label stack, it is delivered to a local software module for processing. Packet forwarding is determined by the label beneath it in the stack. However, if the packet is further forwarded, the router alert label is pushed back onto the label stack before forwarding. The use of this label is analogous to the use of the router

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

alert option in IP packets. Because this label cannot typically occur at the bottom of the stack, it is not associated with a specific network layer protocol.

Note: The router alert label can occur at the bottom of the stack when some operations, administration, and management (OAM) tools are used.

- A value of 3 represents the implicit null label. It is a label that a Label Switching Router (LSR) can assign and distribute, but which does not appear in the encapsulation. When an LSR would otherwise replace the label at the top of the stack with a new label, but the new label is implicit null, the LSR pops the stack instead of doing the replacement. Although this value may never appear in the encapsulation, it needs to be specified in the LDP protocol, so a value is reserved.
- A value of 7 represents the Entropy Label Indicator (ELI) which precedes the actual Entropy Label (EL) that carries the entropy value of the packet in the label stack.
- Values 4 to 6 and 8 to 31 are reserved for future use.

The router uses labels for MPLS, LDP, BGP Label Unicast, Segment Routing, as well as packet-based services such as VLL and VPLS.

To enable SR-MPLS features on SR Linux nodes, you must allocate a label range to the Segment Routing Global Block (SRGB). Other applications that require labels, such as LDP, can then be configured with a dynamic label block.

4.9.1 Configuring null label for static tunnels

About this task

For dynamic tunnels, SR Linux nodes honor reception of null label advertisement, meaning that the nodes push a null label when transmitting (or with implicit null, skip the push of the outermost tunnel label). However, SR Linux nodes do not support advertisement of null labels with dynamic tunnels.

For static tunnels, SR Linux supports the push of null labels using the **nexthop pushed-mpls-label-stack** command.

Note: Null labels are supported on 7250 IXR platforms.

Procedure

To enable **IPV4_EXPLICIT_NULL**, **IPV6_EXPLICIT_NULL**, or **IMPLICIT_NULL** for a next-hop group, use the **pushed-mpls-label-stack** command.

The explicit null label option indicates that the label stack must be popped, and the packet forwarding must be based on the IP header.

The implicit null label option allows an egress LER to receive MPLS packets from the previous hop without the outer LSP label. The operation of the previous hop is referred to as penultimate hop popping (PHP). This option is signaled by the egress LER to the previous hop during the FEC signaling by the LDP control protocol.

In addition, the egress LER can be configured to receive MPLS packets with the implicit null label on a static LSP. LDP must be shut down before changing the implicit null configuration option.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Example: Enable IPv4 explicit null label for next-hop group

The following example enables IPv4 explicit null labels for next-hop group `null-group`.

```
--{ candidate shared default }--[  ]--
# info network-instance default next-hop-groups group null-group
    network-instance default {
        next-hop-groups {
            group null-group {
                nexthop 1 {
                    resolve false
                    pushed-mpls-label-stack [
                        IPV4_EXPLICIT_NULL
                    ]
                }
            }
        }
    }
```

4.10 MPLS ECMP

The following sections describe MPLS ECMP options for SR Linux.

4.10.1 MPLS LSR ECMP (7250 IXR platforms)

SR Linux supports two options to perform MPLS ECMP packet hashing:

- LSR label-only hash (default)
- LSR label hash with Ethernet or IP headers (with L4 and TEID)

LSR label-only hash

By default, MPLS packet hashing at an LSR is based on the whole label stack, along with the incoming port and system IP address. With this label-only hash option, the system hashes the packet using up to 14 labels in the MPLS stack.

The system uses the net result to select which LSP next-hop to send the packet to using a modulo operation of the hash result with the number of next-hops.

This same result feeds to a second round of hashing if there is a LAG on the egress port where the selected LSP has its NHLFE programmed.

LSR label hash with Ethernet or IP headers (with L4 and TEID)

With the LSR label hash option with Ethernet or IP headers (with L4 and TEID), the datapath performs Ethernet speculation after the bottom of the label stack or when the maximum label stack depth is reached. Regardless of the outcome of the Ethernet speculation, the system also attempts IPv4/IPv6 speculation by checking the first nibble after the bottom of the label stack to include the following in the hash:

- source and destination IP address
- UDP/TCP source and destination ports
- GPRS tunneling protocol (GTP) tunnel endpoint identifier (TEID) when the UDP or TCP port is 2123 or 2152

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

If Ethernet header speculation is successful, source and destination MAC are included in the hash along with up to two VLANs.

4.10.2 Configuring MPLS ECMP hashing options

Procedure

To configure MPLS ECMP hashing options on an LSR, use the **system load-balancing lsr-profile** command.

Example: Enable the label-only hash option

The following example enables the label-only hash option on an LSR.

```
--{ candidate shared default }--[  ]--
# system load-balancing lsr-profile label-stack
```

Example: Enable the LSR label hash option with Ethernet or IP headers (with L4 and TEID)

The following example enables the LSR label option with Ethernet or IP headers (with L4 and TEID) on an LSR.

```
--{ candidate shared default }--[  ]--
# system load-balancing lsr-profile label-eth-or-ip-l4-teid
```

4.11 Show reports for MPLS tunnel tables

You can issue a **show** command to display information about MPLS tunnel table entries. You can adjust the output of the report to filter by address type, encapsulation type, tunnel type, and destination prefix.

The following is an example of output from the **show** report.

Output example

```
--{ * candidate shared default }--[  ]--
# show network-instance default tunnel-table all
-------------------------------------------------------------------------------------------------------
IPv4 tunnel table of network-instance "base"
-------------------------------------------------------------------------------------------------------
--
+-------------+-------+-------------+-----------+-----+--------+------+----------------+----------+
| IPv4 Prefix | Encap | Tunnel Type | Tunnel ID | FIB | Metric | Pref | Next-hop (Type)| Next-hop |
|             | Type  |             |           |     |        |      |                |          |
+=============+=======+=============+===========+=====+========+======+================+==========+
| 1.1.1.3/32  | mpls  | ldp         | 65629     | Y   | 2000   | 9    | 1.3.2.3 (mpls) | lag1.1   |
|             |       |             |           |     |        |      | 1.3.3.3 (mpls) | lag1.2   |
|             |       |             |           |     |        |      | 1.3.4.3 (mpls) | lag1.3   |
|             |       |             |           |     |        |      | 1.3.5.3 (mpls) | lag1.4   |
| 1.1.1.4/32  | mpls  | ldp         | 65631     | Y   | 2006   | 9    | 1.7.1.4 (mpls) | lag7.1   |
|             |       |             |           |     |        |      | 1.7.2.4 (mpls) | lag7.2   |
|             |       |             |           |     |        |      | 1.7.3.4 (mpls) | lag7.3   |
|             |       |             |           |     |        |      | 1.7.4.4 (mpls) | lag7.4   |
| 1.1.1.5/32  | mpls  | ldp         | 65630     | Y   | 2007   | 9    | 1.8.2.5 (mpls) | lag8.1   |
|             |       |             |           |     |        |      | 1.8.3.5 (mpls) | lag8.2   |
|             |       |             |           |     |        |      | 1.8.4.5 (mpls) | lag8.3   |
|             |       |             |           |     |        |      | 1.8.5.5 (mpls) | lag8.4   |
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
+-------------+-------+-------------+-----------+-----+--------+------+----------------+----------+
```

4.12 TTM preferences for supported tunnel types

The following table summarizes the TTM preferences for all supported tunnel types on the 7250 IXR systems.

Table 5: TTM preferences for supported tunnel types

Tunnel type	TTM preference
Uncolored SR-MPLS TE-Policy	8
LDP	9
SR-ISIS	11
BGP (LU)	12
Colored SR-MPLS TE-Policy	14

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

5 Configuring LDP

This chapter provides information about configuring Label Distribution Protocol (LDP) on SR Linux for both IPv4 and IPv6.

5.1 Enabling LDP

Procedure

You must enable LDP for the protocol to be active. This procedure applies for both IPv4 and IPv6 LDP.

Example: Enable LDP

The following example administratively enables LDP for the default network instance.

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp admin-state
    network-instance default {
        protocols {
            ldp {
                admin-state enable
            }
        }
    }
```

5.2 Configuring LDP neighbor discovery

Procedure

You can configure LDP neighbor discovery, which allows SR Linux to discover and connect to IPv4 and IPv6 LDP peers without manually specifying the peers. SR Linux supports basic LDP discovery for discovering LDP peers, using multicast UDP hello messages.

Example: Configure LDP neighbor discovery

The following example configures LDP neighbor discovery for a network instance and enables it on a subinterface for IPv4 and IPv6. The **hello-interval** parameter specifies the number of seconds between LDP link hello messages. The **hello-holdtime** parameter specifies how long the LDP link hello adjacency is maintained in the absence of link hello messages from the LDP neighbor.

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp discovery
    network-instance default {
        protocols {
            ldp {
                discovery {
                    interfaces {
                        hello-holdtime 30
                        hello-interval 10
                        interface ethernet-1/1.1 {
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
                            ipv4 {
                                admin-state enable
                            }
                            ipv6 {
                                admin-state enable
                            }
                        }
                    }
                }
            }
        }
    }
```

5.3 Configuring LDP peers

Procedure

You can configure settings that apply to connections between SR Linux and IPv4 and IPv6 LDP peers, including session keepalive parameters. For individual LDP peers, you can configure the maximum number of FEC-label bindings that can be accepted by the peer.

If LDP receives a FEC-label binding from a peer that puts the number of received FECs from this peer at the configured FEC limit, the peer is put into overload. If the peer advertised the Nokia-overload capability (if it is another SR Linux router or an SR OS device) then the overload TLV is transmitted to the peer and the peer stops sending any further FEC-label bindings. If the peer did not advertise the Nokia-overload capability, then no overload TLV is sent to the peer. In either case, the received FEC-label binding is deleted.

Example: Configure LDP peers

The following example configures settings for IPv4 and IPv6 LDP peers:

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp peers
    network-instance default {
        protocols {
            ldp {
                peers {
                    session-keepalive-holdtime 240
                    session-keepalive-interval 90
                    peer 10.1.1.1 label-space-id 0 {
                        fec-limit 1024
                    }
                    peer 2001:db8::0a01:0101 label-space-id 0 {
                        fec-limit 1024
                    }
                }
            }
        }
    }
```

In this example, the **session-keepalive-holdtime** parameter specifies the number of seconds an LDP session can remain inactive (no LDP packets received from the peer) before the LDP session is terminated and the corresponding TCP session closed. The **session-keepalive-interval** parameter specifies the number of seconds between LDP keepalive messages. SR Linux sends LDP keepalive messages at this interval only when no other LDP packets are transmitted over the LDP session.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

For an individual LDP peer, indicated by its LSR ID and label space ID, a FEC limit is specified. SR Linux deletes FEC-label bindings received from this peer beyond this limit.

5.4 Configuring a label block for LDP

Procedure

To configure LDP, you must specify a reference to a predefined range of labels, called a label block. A label block configuration includes a start-label value and an end-label value. LDP uses labels in the range between the start-label and end-label in the label block.

A label block can be static or dynamic. See Static and dynamic label blocks for information about each type of label block and how to configure them. LDP requires a dynamic, non-shared label block.

Example: Configure dynamic LDP label block

The following example configures LDP to use a dynamic label block named `d1`. The dynamic label block is configured with a start-label value and an end-label value. See Configuring label blocks for an example of a dynamic label block configuration.

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp dynamic-label-block
    network-instance default {
        protocols {
            ldp {
                dynamic-label-block d1
            }
        }
    }
```

5.5 Configuring implicit null label

About this task

Note: LDP implicit null labels are supported on 7250 IXR platforms.

The implicit null option allows an egress LER to signal the penultimate (second-last) hop to remove the outer LSP label before forwarding LDP packets to the egress LER, a process referred to as penultimate hop popping (PHP). PHP reduces the processing load on the egress LER because it no longer needs to perform a lookup on the outer LSP label before processing the inner payload.

Procedure

To globally enable the use of the implicit null option for all LDP FECs for which this node is the egress LER, use the **protocols ldp null-label implicit** command. When you change the implicit null option, LDP withdraws all FECs and readvertises them using the new label value.

Example: Configure implicit null label

The following example enables implicit null label for all LDP FECs.

```
--{ candidate shared default }--[  ]--
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
# info network-instance default protocols ldp null-label
    network-instance default {
        protocols {
            ldp {
                null-label implicit
            }
        }
    }
```

5.6 Configuring longest-prefix match for IPv4 and IPv6 FEC resolution

Procedure

By default, SR Linux supports /32 IPv4 and /128 IPv6 FEC resolution using IGP routes. You can optionally enable longest-prefix match for IPv4 and IPv6 FEC resolution. When this is enabled, IPv4 and IPv6 prefix FECs can be resolved by less-specific routes in the route table, as long as the prefix bits of the route match the prefix bits of the FEC. The IP route with the longest prefix match is the route that is used to resolve the FEC.

Example: Configure longest-prefix match for IPv4 and IPv6 FEC resolution

The following example enables longest-prefix match for IPv4 and IPv6 FEC resolution.

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp fec-resolution
    network-instance default {
        protocols {
            ldp {
                fec-resolution {
                    longest-prefix true
                }
            }
        }
    }
```

5.7 Configuring load balancing over equal-cost paths

Procedure

ECMP support for LDP on SR Linux performs load balancing for LDP-based tunnels by using multiple outgoing next-hops for an IP prefix on ingress and transit LSRs. You can specify the maximum number of next-hops (up to 64) to be used for load balancing toward a specific FEC.

Example: Configure load balancing for LDP

The following example configures the maximum number of next-hops that SR Linux can use for load balancing toward an IPv4 or IPv6 FEC.

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp multipath
    network-instance default {
        protocols {
            ldp {
                multipath {
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
                    max-paths 64
                }
            }
        }
    }
```

5.8 Configuring graceful restart helper capability

Procedure

Graceful restart allows a router that has restarted its control plane but maintained its forwarding state to restore LDP with minimal disruption.

To do this, the router relies on LDP peers, which have also been configured for graceful restart, to maintain forwarding state while the router restarts. LDP peers configured in this way are known as graceful restart helpers.

You can configure SR Linux to operate as a graceful restart helper for LDP. When the graceful restart helper capability is enabled, SR Linux advertises to its LDP peers by carrying the fault tolerant (FT) session TLV in the LDP initialization message, which assists the LDP peer to preserve its LDP forwarding state across the restart.

Example: Configure graceful restart

The following example enables the graceful restart helper capability for LDP for both IPv4 and IPv6 peers.

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp graceful--restart
    network-instance default {
        protocols {
            ldp {
                graceful-restart {
                    helper-enable true
                    max-reconnect-time 180
                    max-recovery-time 240
                }
            }
        }
    }
```

In this example, the **max-reconnect-time** parameter specifies the number of seconds the SR Linux waits for the remote LDP peer to reconnect after an LDP communication failure. The **max-recovery-time** parameter specifies the number of seconds the SR Linux router preserves its MPLS forwarding state after receiving the LDP initialization message from the restarted LDP peer.

5.9 Configuring LDP-IGP synchronization

Procedure

You can configure synchronization between LDP and IPv4 or IPv6 interior gateway protocols (IGPs). LDP-IGP synchronization is supported for IS-IS.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

When LDP-IGP synchronization is configured, LDP notifies the IGP to advertise the maximum cost for a link in the following scenarios: when the LDP hello adjacency goes down, when the LDP session goes down, or when LDP is not configured on an interface.

The following apply when LDP-IGP synchronization is configured:

- If a session goes down, the IGP increases the metric of the corresponding interface to max-metric.
- When the LDP adjacency is reestablished, the IGP starts a hold-down timer (default 60 seconds). When this timer expires, the IGP restores the normal metric, if it has not been restored already.
- When LDP informs the IGP that all label-FEC mappings have been received from the peer, the IGP can be configured to immediately restore the normal metric, even if time remains on the hold-down timer.

LDP-IGP synchronization does not take place on LAN interfaces unless the IGP has a point-to-point connection over the LAN, and does not take place on IGP passive interfaces.

Example: Configure LDP-IGP synchronization

The following example configures LDP-IGP synchronization for an IS-IS instance.

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols isis instance i1 ldp-synchronization
    network-instance default {
        protocols {
            isis {
                instance i1 {
                    ldp-synchronization {
                        hold-down-timer 120
                        end-of-lib true
                    }
                }
            }
        }
    }
```

In this example, if the LDP adjacency goes down, the IGP increases the metric of the corresponding interface to max-metric. If the adjacency is subsequently reestablished, the IGP waits for the amount of time configured with the **hold-down-timer** parameter before restoring the normal metric.

When the **end-of-lib** parameter is set to **true**, it causes the IGP to restore the normal metric when all label-FEC mappings have been received from the peer, even if time remains in the hold-down timer.

5.10 Configuring LDP tunnel-down damp time

About this task

The **tunnel-down-damp-time** command specifies the time interval in seconds that LDP waits before posting a tunnel down event to the Tunnel Table Manager (TTM). The default value is 3 seconds.

When LDP can no longer resolve a FEC and deactivates it, it deprograms the NHLFE in the datapath. However, LDP delays deleting the LDP tunnel entry in the TTM until the **tunnel-down-damp-time** timer expires. This means users of the LDP tunnel, such as BGP (Layer 3 VPN) are not immediately notified. Traffic is therefore blackholed temporarily because the forwarding engine NHLFE has been deprogrammed.

If the FEC is resolved before the **tunnel-down-damp-time** timer expires, then LDP programs the forwarding engine with the new NHLFE and performs a tunnel modify event, updating the dampened entry

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

in the TTM with the new NHLFE information. If the FEC does not get resolved and the **tunnel-down-damp-time** timer expires, LDP posts a tunnel down event to TTM which deletes the LDP tunnel.

When there is an upper layer user of LDP that depends on the LDP control plane for failover detection, then the **label-withdrawal-delay** and **tunnel-down-damp-time** commands must be set to 0. For example, with a pseudowire redundancy, the primary pseudowire does not have its own fast failover detection mechanism and the node depends on the LDP tunnel down event to activate the standby pseudowire.

Procedure

To configure the tunnel-down damp time, use the **ldp tunnel-down-damp-time** command.

Example: Configure LDP tunnel-down damp time

The following example sets the LDP tunnel-down damp time to 5 seconds.

```
--{ candidate shared default }--[  ]--
# info network-instance default protocols ldp tunnel-down-damp-time
    network-instance default {
        protocols {
            ldp {
                tunnel-down-damp-time 5
            }
        }
    }
```

5.11 Configuring LDP label withdrawal delay

About this task

When an LDP FEC is deactivated, the **label-withdrawal-delay** command specifies the time interval in seconds that LDP delays the withdrawal of the associated FEC-label binding that was distributed to its neighbors. When the timer expires, LDP sends a label withdrawal for the FEC to all its neighbors. This feature is applicable only to LDP IPv4 and IPv6 prefix FECs and is not applicable to pseudowires (service FECs).

When there is an upper layer user of LDP that depends on the LDP control plane for failover detection, then the **label-withdrawal-delay** and **tunnel-down-damp-time** commands must be set to 0. For example, with a pseudowire redundancy, the primary pseudowire does not have its own fast failover detection mechanism and the node depends on the LDP tunnel down event to activate the standby pseudowire.

Procedure

To configure the LDP label withdrawal delay, use the **ldp label-withdrawal-delay** command.

Example: Configure LDP label withdrawal delay

The following example sets the LDP label withdrawal delay to 20 seconds.

```
--{ candidate shared default }--[  ]--
# info network-instance default protocols ldp label-withdrawal-delay
    network-instance default {
        protocols {
            ldp {
                label-withdrawal-delay 20
            }
        }
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
    }
```

5.12 Configuring static FEC (FEC originate)

About this task

Static FEC (also known as FEC originate) triggers a label for prefix announcement without a requirement to enable LDP on a subinterface or to receive a label from a neighbor. Network security requirements may dictate that prefixes advertised using LDP must not be directly associated with a subinterface or system IP address. Static FEC allows the router to advertise these prefixes and their labels to peers as LDP FECs to provide reachability to the desired IP addresses or subnets without explicitly using a subinterface address. The router can advertise a FEC with a pop action.

A FEC can be added to the LDP IP prefix database with a specific label operation on the node. The only permitted operation is pop (the **swap** parameter must be set to **false**).

A route-table entry is required for a FEC to be advertised. Static FEC is supported for both IPv4 and IPv6 FECs.

Procedure

To configure static FEC, use the **static-fec** command under the **network-instance protocols ldp** context.

Example: Configure static FEC

```
--{ + candidate shared default }--[  ]--
# info network-instance default protocols ldp static-fec 10.10.10.2/32
    network-instance default {
        protocols {
            ldp {
                static-fec 10.10.10.2/32 {
                    swap false
                }
            }
        }
    }
```

5.13 Overriding the LSR ID on an interface

About this task

For security concerns, it may be beneficial to overwrite the default LSR ID with a different LSR ID for link or targeted LDP sessions. The following LSR IDs can be overwritten:

- IPv4 l-LDP: local subinterface IPv4 address
- IPv6 l-LDP: local subinterface IPv4 or IPv6 address, in one of the following combinations:
  - the subinterface IPv4 address as the 32-bit LSR-ID and the subinterface IPv6 address as the transport connection address
  - the subinterface IPv6 address as both a 128-bit LSR-ID and transport connection address
- IPv4 T-LDP: IPv4 loopback or any IPv4 LDP subinterface
- IPv6 T-LDP: IPv4 loopback, IPv6 loopack, IPv4 LDP subinterface, or IPv6 LDP subinterface

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Note that a loopback interface cannot be used with the 32-bit format.

To change the value of the LSR ID on an interface to the local interface IP address, use the **override-lsr-id** option. When **override-lsr-id** is enabled, the transport address for the LDP session and the source IP address of the Hello messages is updated to the interface IP address.

In IPv6 networks, either the IPv4 or IPv6 interface address can override the IPv6 LSR ID.

Procedure

To override the value of the LSR ID on an interface, use the following commands under the **network-instance** context:

- **IPv4**:

  protocols ldp discovery interfaces interface <*name*> **ipv4 override-lsr-id local-subinterface ipv4**

- **IPv6**:

  protocols ldp discovery interfaces interface <*name*> **ipv6 override-lsr-id local-subinterface [ipv4 | ipv6]**

Example: Configure the LSR ID

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp discovery interfaces interface ethernet-1/
1.1
    network-instance default {
        protocols {
            ldp {
                discovery {
                    interfaces {
                        interface ethernet-1/1.1 {
                            ipv4 {
                                override-lsr-id {
                                    local-subinterface ipv4
                                }
                            }
                            ipv6 {
                                override-lsr-id {
                                    local-subinterface ipv6
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

5.14 LDP FEC import and export policies

SR Linux supports FEC prefix import and export policies for LDP, which provide filtering of both inbound and outbound LDP label bindings.

FEC prefix export policy

A FEC prefix export policy controls the set of LDP prefixes and associated LDP label bindings that a router advertises to its LDP peers. By default, the router advertises local label bindings for only the

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

system address, but propagates all FECs that are received from neighbors. The export policy can also be configured to advertise local interface FECs.

The export policy can accept or reject label bindings for advertisement to the LDP peers.

When applied globally, LDP export policies apply to FECs advertised to all neighbors. To control the propagation of FECs to a specific LDP neighbor, you can apply an LDP export prefix policy to the specified peer.

Note: The export policy does not support blocking of static FECs.

FEC prefix import policy

A FEC prefix import policy controls the set of LDP prefixes and associated LDP label bindings received from other LDP peers that a router accepts. The router redistributes all accepted LDP prefixes that it receives from its neighbors to other LDP peers (unless rejected by the FEC prefix export policy).

The import policy can accept or reject label bindings received from LDP peers. By default, the router imports all FEC prefixes from its LDP peers.

When applied globally, LDP import policies apply to FECs received from all neighbors. To control the import of FECs from a specific LDP neighbor, you can apply an LDP import prefix policy to the specified peer.

Routing policies

The filtering of label bindings in LDP FEC import and export policies is based on prefix lists that are defined using routing policies.

5.14.1 Configuring routing policies for LDP FEC import and export policies

Procedure

To configure routing policies for LDP FEC import and export policies, use the **routing-policy** command.

Example: Configure routing policy for global LDP FEC import and export

The following shows an example configuration of global LDP FEC import and export policies, defining match rules to accept an import prefix set, and reject an export prefix set.

```
--{ * candidate shared default }--[ ]--
# info routing-policy
    routing-policy {
        prefix-set export-prefix-set-test {
            prefix 10.1.1.2/32 mask-length-range exact {
            }
            prefix 10.1.1.3/32 mask-length-range exact {
            }
        }
        prefix-set import-prefix-set-test {
            prefix 10.1.1.1/32 mask-length-range exact {
            }
            prefix 10.1.1.2/32 mask-length-range exact {
            }
            prefix 10.1.1.3/32 mask-length-range exact {
            }
        }
        policy export-fec-test {
            default-action {
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
                policy-result accept
            }
            statement export-statement-test {
                match {
                    prefix-set export-prefix-set-test
                }
                action {
                    policy-result reject
                }
            }
        }
        policy import-fec-test {
            default-action {
                policy-result accept
            }
            statement import-statement-test {
                match {
                    prefix-set import-prefix-set-test
                }
                action {
                    policy-result accept
                }
            }
        }
    }
```

Example: Configure routing policy for peer LDP FEC import and export

The following shows an example configuration of peer LDP FEC import and export policies, defining match rules to reject an import prefix set and an export prefix set for the specified peer.

```
--{ * candidate shared default }--[ ]--
# info routing-policy
    routing-policy {
        prefix-set peer-export-prefix-test {
            prefix 10.1.1.4/32 mask-length-range exact {
            }
        }
        prefix-set peer-import-prefix-test {
            prefix 10.1.1.5/32 mask-length-range exact {
            }
        }
        policy peer-export-test {
            statement peer-export-statement-test {
                match {
                    prefix-set export-prefix-set-test
                }
                action {
                    policy-result reject
                }
            }
        }
        policy peer-import-test {
            statement peer-import-statement-test {
                match {
                    prefix-set import-prefix-set-test
                }
                action {
                    policy-result reject
                }
            }
        }
    }
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

5.14.2 Applying global LDP FEC import and export policies

Procedure

Use the **ldp import-prefix-policy** and **ldp export-prefix-policy** commands to apply global LDP FEC import and export policies.

The following example applies global export and import policies to LDP FECs.

Example: Apply global LDP import and export policies

```
--{ +* candidate shared default }--[  ]--
# info network-instance default protocols ldp
    network-instance default {
        protocols {
            ldp {
                export-prefix-policy export-fec-test
                import-prefix-policy import-fec-test
            }
        }
    }
```

5.14.3 Applying per-peer LDP FEC import and export policies

Procedure

Use the **import-prefix-policy** and **export-prefix-policy** commands under the **ldp peers peer** context to apply per-peer LDP FEC import and export policies.

The following example applies LDP FEC export and import policies to LDP peer 10.10.10.1.

Example: Apply global LDP import and export policies

```
--{ +* candidate shared default }--[  ]--
# info network-instance default protocols ldp peers peer 10.10.10.1 label-space-id 1
    network-instance default {
        protocols {
            ldp {
                peers {
                    peer 10.10.10.1 label-space-id 1 {
                        export-prefix-policy peer-export-test
                        import-prefix-policy peer-import-test
                    }
                }
            }
        }
    }
```

5.15 Targeted LDP

Note: Targeted LDP (T-LDP) is supported on 7250 IXR Gen 2c+ and 7250 IXR Gen 3, platforms only. All of these platforms support tT-LDP for computation of P,Q space for remote LFA.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

SR Linux supports targeted LDP (T-LDP) for both IPv4 and IPv6, using IS-IS or OSPF/OSPFv3 as the IGP. The following table shows the feature support by platform.

Table 6: T-LDP IPv4/IPv6 support by platform

T-LDP IGP	7250 IXR Gen 2c+	7250 IXR Gen 3
T-LDP v4, IS-IS	supported	supported
T-LDP v6, IS-IS	—	—
T-LDP v4, OSPF	supported	supported
T-LDP v6, OSPFv3	—	—

While LDP sessions are typically established between two directly connected peers, T-LDP allows LDP sessions to be established between non-adjacent peers using LDP extended discovery. The source and destination addresses of targeted Hello packets are the LSR-IDs of the remote LDP peers.

In SR Linux, T-LDP sessions use the system IP address as the LSR-ID by default. You can optionally configure the **override-lsr-id** parameter on the targeted session to change the value of the LSR-ID to either an LDP-enabled local subinterface or to a loopback subinterface. The transport address for the LDP session and the source IP address of the targeted Hello message are then updated to the new LSR-ID value.

With IPv4 T-LDP, the **override-lsr-id** parameter can update the LSR-ID to an IPv4 subinterface or loopback subinterface. With IPv6 T-LDP, the **override-lsr-id** parameter can update the LSR-ID to an IPv4 or IPv6 subinterface or loopback subinterface.

5.15.1 Configuring targeted LDP

Procedure

To configure an IPv4 or IPv6 targeted LDP session, use the **network-instance protocols ldp discovery targeted** command.

Example: Configure targeted LDP

The following example shows an IPv4 targeted LDP session with BFD and advertise FEC enabled, custom hello holdtime and interval values defined, and the LSR ID set to the IPv4 address for subinterface ethernet-1/10.1.

```
--{ * candidate shared default }--[  ]--
# info network-instance default protocols ldp discovery targeted
    network-instance default {
        protocols {
            ldp {
                discovery {
                    targeted {
                        ipv4 {
                            target 10.0.0.1 {
                                admin-state enable
                                enable-bfd true
                                advertise-fec true
                                hello-holdtime 30
                                hello-interval 10
                                override-lsr-id {
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

```
                                    subinterface-ipv4 ethernet-1/10.1
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

5.16 Manual LDP RLFA

LDP remote LFA (RLFA) builds on the existing segment routing (SR) RLFA capability to compute repair paths to a remote LFA node (or PQ node). After a link failure, RLFA puts packets back onto the shortest path without looping them back to the node that forwarded them. See the *SR Linux Segment Routing Guide* for more information about RLFA computation.

Unlike automatic LDP RLFA, manual LDP RLFA requires the targeted sessions (between source node and PQ node) to be manually configured before a link failure. RLFA repair tunnels can be LDP tunnels only.

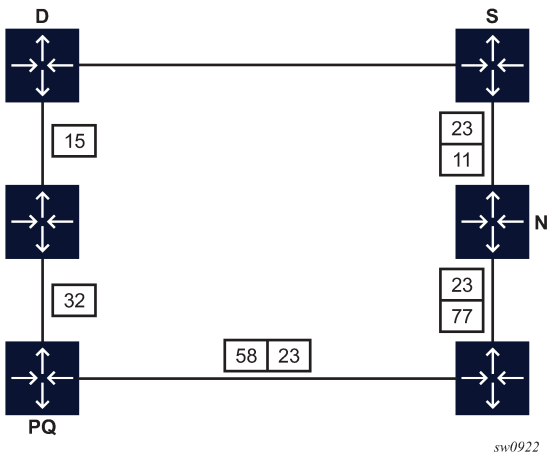
Supported platforms

Manual LDP RLFA is supported on 7250 IXR Gen 2c+ and 7250 IXR Gen 3, platforms only.

General principles of LDP RLFA

The following figure shows the general principles of LDP RLFA operation using an LDP-in-LDP repair tunnel.

Figure 2: General principles of LDP RLFA operation



In this figure, S is the source node and D is the destination node. The primary path is the direct link between nodes S and D, and the RLFA algorithm on node S determines the PQ node for destination D. In the event of a failure between nodes S and D, to prevent traffic from looping back to node S, the traffic must be sent to the PQ node. A targeted LDP (T-LDP) session is required between nodes PQ and S. Over that T-LDP session, the PQ node advertises label 23 for FEC D. All other labels are link LDP bindings, which allow traffic to reach the PQ node. On node S, LDP creates an NHLFE that has two labels, where

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

label 23 is the inner label. Label 23 is tunneled to the PQ node, which then forwards traffic on the shortest path to node D.

As prerequisites for RLFA to operate, the LFA algorithm must be run (**loopfree-alternate remote-lfa**), and the PQ node information must also be attached to LDP route entries (**loopfree-alternate augment-route-table**).

Feature considerations

Be aware of the following feature considerations:

- LDP RLFA applies to IPv4 FECs with IS-IS and OSPFv2 only.
- Manual LDP RLFA requires the targeted sessions (between source node and PQ node) to be manually configured before a link failure. The system does not automatically set up T-LDP sessions toward the PQ nodes that the RLFA algorithm has identified. These targeted sessions must be set up with router IDs that match the IDs the RLFA algorithm uses.
- LDP RLFA is designed to operate in LDP-only environments.
- The **lsp-trace** OAM command is not supported over the repair tunnels.

5.16.1 Configuring manual LDP RLFA

Prerequisites

In IS-IS:

- Enable remote LFA computation using the following command:

  network-instance protocols isis instance loopfree-alternate remote-lfa admin-state enable
- Enable attaching RLFA information to RTM entries using the following command:

  network-instance protocols isis instance loopfree-alternate augment-route-table true

  This second command attaches RLFA-specific information to route entries that are necessary for LDP to program repair tunnels toward the PQ node using a specific neighbor.

Or, in OSPFv2:

- Enable remote LFA computation using the following command:

  network-instance protocols ospf instance loopfree-alternate remote-lfa admin-state enable
- Enable attaching RLFA information to RTM entries using the following command:

  network-instance protocols ospf instance loopfree-alternate augment-route-table true

  This second command attaches RLFA-specific information to route entries that are necessary for LDP to program repair tunnels toward the PQ node using a specific neighbor.

Procedure

Step 1. To enable manual LDP RLFA on the source node and PQ node, configure targeted LDP sessions using the **network-instance protocols ldp discovery targeted target** command.

Step 2. On the PQ node, enable FEC advertisement using the **network-instance protocols ldp discovery targeted [ipv4 | ipv6] target advertise-fec** command.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Example: Configure manual LDP RLFA on a source node

The following example configures manual LDP RLFA on a source node, with no **advertise-fec** configuration.

```
--{ candidate shared default }--[  ]--
# info network-instance default protocols ldp discovery targeted
    network-instance default {
        protocols {
            ldp {
                discovery {
                    targeted {
                        ipv4 {
                            target 10.1.1.10 {
                                admin-state enable
                                hello-holdtime 60
                                hello-interval 30
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Configure manual LDP RLFA on a PQ node

The following example configures manual LDP RLFA on a PQ node, with **advertise-fec** set to **true**.

```
--{ candidate shared default }--[  ]--
# info network-instance default protocols ldp discovery targeted
    network-instance default {
        protocols {
            ldp {
                discovery {
                    targeted {
                        ipv4 {
                            target 10.1.1.15 {
                                admin-state enable
                                advertise-fec true
                                hello-holdtime 60
                                hello-interval 30
                            }
                        }
                    }
                }
            }
        }
    }
```

5.17 Automatic LDP RLFA

Unlike manual LDP RLFA, automatic LDP RLFA automatically establishes targeted LDP sessions without the need to specify a list of peers that require targeted sessions. On the source node, specifying which peer serves as the PQ node is not required. Instead, with automatic RLFA, the source node automatically identifies and establishes targeted LDP sessions with the PQ node. The PQ node then provides an LDP-in-LDP repair path to the destination following a failure. The automatic LDP RLFA feature minimizes overall configuration, providing a more dynamic and flexible environment.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

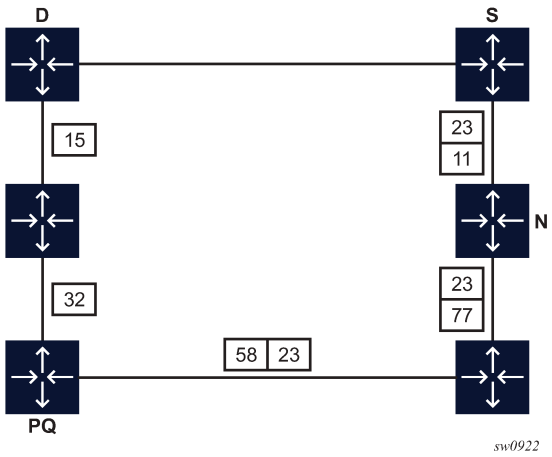
Supported platforms

Automatic LDP RLFA is supported on 7250 IXR Gen 2c+, and 7250 IXR Gen 3, platforms only.

Simple scenario: one PQ node and one S node

The basic principles of operation for the automatic LDP RLFA capability are the same as in manual LDP RLFA. The following figure again shows the general principles of LDP RLFA operation.

Figure 3: General principles of LDP RLFA operation



In this example, to address a failure on the shortest path between nodes S and D, node S needs a targeted LDP session toward the PQ node to learn the label-binding information on the PQ node for FEC D. To enable automatic RLFA, the same LFA algorithm used for manual RLFA must be run (**loopfree-alternate remote-lfa**), and the PQ node information must also be attached to LDP route entries (**loopfree-alternate augment-route-table**).

Because node S is the node that requires the T-LDP session, it is configured as the transmitting node (**auto-tx**) to initiate the T-LDP session. The PQ node is configured as the receiving node (**auto-rx**) of the targeted hello request for that session. And similar to manual LDP RLFA, the PQ node requires FEC advertisement to be enabled (**auto-rx advertise-fec enable**) so that the PQ node can send to the S node the label it has bound to FEC D (label 23).

With automatic RLFA enabled, node S uses the PQ node information attached to the route entries to automatically start sending targeted hellos to the PQ node. The PQ node accepts those hellos and the T-LDP session is established.

If the network topology changes and the PQ node of S for FEC D changes, S automatically brings down the T-LDP session to the previous PQ node and attempts to establish a new one towards the new PQ node.

Typical scenario: nodes are both PQ node and S node

In the previous simple example of one S node and one PQ node, **auto-rx** and FEC advertisement can remain disabled on the S node. However, in typical network deployments, each node is potentially the source node and also the PQ node for a given destination FEC. Therefore, all nodes can have both **auto-tx** and **auto-rx** enabled. These nodes can also have other configurations defined for specific LDP peers. These multiple configurations (either explicit or implicit) have implications for the operation of LDP RLFA.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

The first implication relates to the fact that LDP operates using precedence levels. When a targeted session is established with a peer, LDP uses the session parameters with the highest precedence. The order of precedence is as follows (in descending order of priority):

1. **peer**
2. **auto-tx**
3. **auto-rx**

To illustrate how LDP precedence affects LDP RLFA, consider a T-LDP session between the following nodes:

- node A (source node)
- node B (PQ node)

If source node A has both a per-peer configuration for node B and **auto-tx** enabled, node A chooses the parameters defined in the per-peer configuration rather than those defined under **auto-tx** to establish the session. A similar precedence also applies on PQ node B. However, when node B uses the per-peer configuration it has for node A rather than the **auto-rx** options (including **advertise-fec enable**), LDP RLFA cannot operate unless the per-peer configuration also enables **advertise-fec**.

A similar logic also applies in that the **auto-tx** configuration takes precedence over **auto-rx**. In a typical scenario, both these modes are enabled on a node that needs to act as both source and PQ node. However, if the node uses the **auto-tx** configuration for the T-LDP session (because it has the highest precedence), **advertise-fec** must be also enabled under **auto-tx** for LDP RLFA to function.

The second implication of multiple LDP configurations is that redundant T-LDP sessions may remain up after a topology change when they are no longer required. In this case, a **tools** command allows you to delete these redundant T-LDP sessions.

Configuration considerations

The following configuration considerations apply:

- automatic LDP RLFA works with IS-IS and OSPFv2 only
- automatic LDP RLFA only supports IPv4 FECs
- **override-lsr-id** configuration is not supported
- **lsp-trace** on the backup path is not supported

5.17.1 Configuring automatic LDP RLFA

Prerequisites

In IS-IS:

- Enable remote LFA computation using the following command:

  network-instance protocols isis instance loopfree-alternate remote-lfa admin-state enable
- Enable attaching RLFA information to RTM entries using the following command:

  network-instance protocols isis instance loopfree-alternate augment-route-table true

Or, in OSPFv2:

- Enable remote LFA computation using the following command:

  network-instance protocols ospf instance loopfree-alternate remote-lfa admin-state enable

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

- Enable attaching RLFA information to RTM entries using the following command:

  network-instance protocols ospf instance loopfree-alternate augment-route-table true

  This second command attaches RLFA-specific information to route entries that are necessary for LDP to program repair tunnels toward the PQ node using a specific neighbor.

Procedure

To enable automatic LDP RLFA, use the **auto-rx** and **auto-tx** options under **network-instance default protocols ldp discovery targeted**, as follows:

- On all potential source nodes, enable **auto-rx**.
- On all potential PQ nodes, enable **auto-tx**.
- On PQ-only nodes, enable the **advertise-fec** option under **auto-rx**.
- On PQ nodes that can also operate as source nodes, enable the **advertise-fec** option under both **auto-rx** and **auto-tx**.

Example: Configure automatic LDP RLFA on node that is both source and PQ node

The following example configures automatic LDP RLFA on a node that is both source and PQ node, with **advertise-fec** set to **true** under both **auto-rx** and **auto-tx**.

```
--{ candidate shared default }--[  ]--
# info network-instance default protocols ldp discovery targeted ipv4
    network-instance default {
        protocols {
            ldp {
                discovery {
                    targeted {
                        ipv4 {
                            auto-rx {
                                admin-state enable
                                advertise-fec true
                            }
                            auto-tx {
                                admin-state enable
                                advertise-fec true
                            }
                        }
                    }
                }
            }
        }
    }
```

5.17.2 Displaying source context for automatic T-LDP session parameters

About this task

It is not possible to configure LDP options specifically for automatic T-LDP sessions except for **advertise-fec**. The session inherits the settings defined for the IPv4 and IPv6 families (under **ldp discovery targeted**) or the system default values. This applies to the following configurations:

- **network-instance default protocols ldp discovery targeted hello-holdtime**
- **network-instance default protocols ldp discovery targeted hello-interval**

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Procedure

To identify the context from which the active T-LDP session configuration options are taken, use the **info from state** command.

Example: Display source context for automatic T-LDP session parameters

```
--{ candidate shared default }--[  ]--
# info from state network-instance default protocols ldp discovery targeted ipv4 target
 10.1.1.6 oper-type
    network-instance base {
        protocols {
            ldp {
                discovery {
                    targeted {
                        ipv4 {
                            target 10.1.1.6 {
                                oper-type auto-tx
                            }
                        }
                    }
                }
            }
        }
    }
```

5.17.3 Deleting redundant T-LDP sessions

Procedure

To delete any redundant T-LDP sessions, use the **tools network-instance default protocols ldp targeted-auto-rx hold-time** command.

You must run the command during a specific time window on all nodes that have **auto-rx** enabled. The **hold-time** value must be greater than the hello-timer value plus the time required to run the **tools** command on all applicable nodes. Ensure that the configured value is long enough to meet these requirements. A system check verifies that a non-zero value is configured, but no other checks are enforced.

Note: The **tools network-instance default protocols ldp targeted-auto-rx hold-time** command is not synchronized to the standby CPM. If you perform the operation with a large hold-time value and the CPM does a switchover during this time, you must restart the operation on the newly active CPM.

Example: Deleting redundant T-LDP sessions

The following example enables the **targeted-auto-rx hold-time** command to delete any redundant T-LDP sessions.

```
--{ candidate shared default }--[  ]--
# tools network-instance default protocols ldp targeted-auto-rx hold-time 120
```


© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Example: Display remaining timeout value

While the **targeted-auto-rx hold-time** command is in progress, the following example shows how to display the remaining timeout value using the **info from state** command.

```
--{ candidate shared default }--[  ]--
# info from state network-instance default protocols ldp discovery targeted ipv4 target
 10.10.10.15 hello-adjacencies adjacency 10.10.10.10 label-space-id 10 hello-holdtime
 remaining
```

5.18 LSP ping and trace for LDP tunnels

To check connectivity and trace the path to any midpoint or endpoint of an LDP tunnel, SR Linux supports the following OAM commands:

- **tools oam lsp-ping ldp fec** <*prefix*>
- **tools oam lsp-trace ldp fec** <*prefix*>

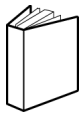
Supported parameters include **destination-ip**, **source-ip**, **timeout**, **ecmp-next-hop-select**, and **traffic-class**. However, the only mandatory parameter is **fec**.

Results from the lsp-ping and lsp-trace operations are displayed using **info from state** commands.

For more information, see the *SR Linux OAM and Diagnostics Guide*.

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Customer document and product support



Customer documentation

Customer documentation welcome page



Technical support

Product support portal



Documentation feedback

Customer documentation feedback