Nokia Service Router Linux
7215 Interconnect System
7220 Interconnect Router
7250 Interconnect Router

Release 26.10

VPN Services Guide

3HE 22254 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 9

1.1 Precautionary and information messages 9

1.2 Conventions 9

2 What's new 11

3 Overview 13

3.1 About EVPN 13

3.2 About the service infrastructure for EVPN 13

3.3 About EVPN for Layer 2 14

3.4 About EVPN for Layer 3 15

3.5 About EVPN for multicast 16

3.6 About IP-VPN 17

4 EVPN services 18

4.1 Layer 2 services infrastructure 18

4.1.1 MAC-VRF network instance 18

4.1.1.1 MAC selection 18

4.1.1.2 MAC duplication detection and actions 18

4.1.1.3 Bridge table configuration 19

4.1.2 Interface extensions for Layer 2 services 20

4.1.2.1 Traffic classification and ingress/egress mapping actions 20

4.1.2.2 Routed and bridged subinterfaces 21

4.1.3 IRB interfaces 21

4.1.3.1 Using ACLs with IRB interfaces and Layer 2 subinterfaces 22

4.1.4 Layer 2 services configuration 22

4.1.4.1 MAC-VRF network instance configuration example 22

4.1.4.2 Bridged subinterface configuration example 23

4.1.4.3 IRB interface configuration example 25

4.1.5 Displaying bridge table information 25

4.1.6 Deleting entries from the bridge table 28

4.1.7 Storm control on bridged subinterfaces 28

4.1.7.1 Configuring storm control for an interface 30

4.1.7.2 Displaying storm control information 31

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

4.1.8 L2CP transparency....32
4.1.8.1 Configuring L2CP transparency....34
4.1.8.2 Configuring dot1x individual tunneling....36
4.1.8.3 Displaying L2CP transparency information....36
4.1.8.4 Displaying L2CP transparency statistics....38
4.1.9 Split horizon groups on MAC-VRF....39
4.1.9.1 Configuring split horizon groups....40
4.1.9.2 Split horizon group configuration example....40
4.1.9.3 Platform specifications for split horizon groups....41
4.1.10 Server aggregation configuration example....42
4.1.10.1 Configuration for server aggregation example....43
4.1.11 Spanning Tree Protocol (STP)....45
4.1.11.1 Configuring STP on a bridged subinterface....47
4.1.11.2 Configuring STP on a MAC-VRF....48
4.1.11.3 Configuring a management STP instance....50
4.1.11.4 Configuring RSTP mode on a management STP instance....54
4.1.11.5 Configuring BPDU guard....54
4.1.11.6 Configuring BPDU guard recovery time....55
4.2 VXLAN v4....55
4.2.1 VXLAN configuration....55
4.2.1.1 Source and destination VTEP addresses....60
4.2.1.2 Ingress or egress VNI....60
4.2.1.3 VLAN tagging for VXLAN....60
4.2.1.4 Network instance and interface MTU....60
4.2.1.5 Fragmentation for VXLAN traffic....60
4.2.2 VXLAN and ECMP....61
4.2.3 VXLAN ACLs....61
4.2.4 QoS for VXLAN tunnels....62
4.2.4.1 Configuring a VXLAN classifier policy....62
4.2.5 Asymmetric VXLAN VNIs....63
4.2.6 VXLAN statistics collection....63
4.2.6.1 Enabling VXLAN statistics collection....64
4.2.6.2 Displaying VXLAN statistics....64
4.2.6.3 Clearing VXLAN statistics....65
4.3 MPLS transport for EVPN services....65
4.4 EVPN for Layer 2 ELAN services....67

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

4.4.1 Basic configuration for EVPN for Layer 2 ELAN services ... 67
4.4.1.1 EVPN Layer 2 basic routes ... 69
4.4.1.2 Creation of EVPN destinations based on received EVPN routes ... 70
4.4.1.3 EVPN route selection ... 73
4.4.1.4 BGP next hop configuration for EVPN routes ... 73
4.4.2 MAC duplication detection for Layer 2 loop prevention in EVPN ... 74
4.4.3 EVPN Layer 2 multihoming ... 74
4.4.3.1 EVPN Layer 2 multihoming procedures ... 75
4.4.3.2 All-active multihoming ... 76
4.4.3.3 Single-active multihoming ... 76
4.4.3.4 Reload-delay timer ... 78
4.4.3.5 EVPN multihoming for IPv6 and non-system IPv4 next-hops ... 80
4.4.3.6 EVPN multihoming configuration example ... 80
4.4.4 Layer 2 proxy-ARP/ND ... 84
4.4.4.1 Dynamic learning for proxy-ARP/ND ... 85
4.4.4.2 Static proxy-ARP entries ... 87
4.4.4.3 EVPN learning for proxy-ARP/ND ... 88
4.4.4.4 Configuring proxy-ARP/ND traffic flooding options ... 89
4.4.4.5 Proxy-ARP/ND duplicate IP detection ... 91
4.4.4.6 Proxy-ARP/ND table ... 94
4.4.4.7 EVPN ARP/ND extended community flags ... 96
4.4.4.8 Displaying proxy-ARP/ND information ... 100
4.4.4.9 Displaying proxy-ARP/ND statistics ... 102
4.4.5 EVPN VLAN-aware bundle services ... 103
4.4.5.1 Configuring the Ethernet Tag ID for VLAN-aware bundle services ... 104
4.4.5.2 Displaying the Ethernet Tag ID for VLAN-aware bundle services ... 105
4.4.6 EVPN Proxy MAC/IP Advertisement routes in MAC-VRFs with proxy-ARP/ND ... 105
4.4.7 Flow-aware transport label and Layer 2 attribute signaling for EVPN in MAC-VRF ... 106
4.5 EVPN for Layer 2 ELINE services ... 108
4.5.1 Configuring EVPN-VPWS services ... 109
4.5.2 Operational state in VPWS network instances ... 110
4.5.3 Layer 2 attribute mismatch ... 111
4.5.4 Flow-aware transport label for EVPN-VPWS ... 111
4.5.5 Interface/LAG link loss forwarding ... 112
4.6 EVPN for Layer 3 ... 113
4.6.1 EVPN Layer 3 basic configuration ... 113

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

4.6.1.1 Asymmetric IRB ... 114
4.6.1.2 Symmetric IRB interface-less ip-vrf-to-ip-vrf model ... 119
4.6.1.3 Symmetric IRB model using EVPN MAC/IP advertisement routes in the route table ... 125
4.6.1.4 EVPN interface-ful model (EVPN IFF) ... 128
4.6.2 Anycast gateways ... 134
4.6.3 EVPN Layer 3 multihoming and anycast gateways ... 136
4.6.4 EVPN Proxy MAC/IP Advertisement routes in MAC-VRFs with IRB interfaces ... 137
4.6.5 EVPN Layer 3 host route mobility ... 140
4.6.6 EVPN-VXLAN IFL interoperability with EVPN IFF ... 143
4.6.7 VIP discovery for redundant servers ... 144
4.6.7.1 Configuring VIP discovery ... 145
4.6.7.2 Displaying VIP discovery information ... 147
4.6.8 Layer 3 proxy-ARP/ND ... 148
4.6.8.1 Layer 3 proxy-ARP ... 149
4.6.8.2 Layer 3 proxy-ND ... 151
4.6.9 EVPN IP aliasing ... 151
4.6.9.1 Configuring Layer 3 Ethernet Segments for EVPN IFL IP aliasing ... 158
4.6.10 Unequal ECMP for EVPN IP prefix routes ... 164
4.6.10.1 Advertising the EVPN link-bandwidth extended community ... 166
4.6.10.2 Enabling weighted ECMP ... 167
4.6.10.3 Configuring weighted ECMP for received PE-CE BGP routes ... 168
4.6.10.4 Displaying ECMP weights ... 168
4.6.11 Resilient hashing for EVPN IFL ... 169
4.6.11.1 Resilient hashing and unequal ECMP for EVPN IFL ... 170
4.6.12 Gateway IP-based load balancing for EVPN IFL routes ... 172
4.7 Dynamic multicast ID management ... 176
4.7.1 Displaying MCID information ... 177
4.8 EVPN for multicast ... 178
4.8.1 IGMP/MLD snooping ... 178
4.8.2 EVPN multicast for Layer 2 ... 179
4.8.2.1 EVPN multihoming and multicast synchronization ... 180
4.8.2.2 Configuring EVPN multicast for Layer 2 ... 181
4.8.2.3 Displaying EVPN multicast for Layer 2 information ... 185
4.8.3 EVPN multicast for Layer 3 (RFC 9625) ... 192
4.8.3.1 Data path requirements ... 193

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

4.8.3.2 Control plane requirements ... 194
4.8.3.3 Configuring EVPN multicast for Layer 3 ... 194
4.8.3.4 PEG function with ingress replication ... 197
4.9 BGP and routing policy extensions for EVPN ... 202
4.9.1 BGP extensions for EVPN ... 202
4.9.2 Best path selection for EVPN IFL routes ... 203
4.9.3 Routing policy extensions for EVPN ... 203
4.9.4 Best path selection for EVPN AD per-EVI routes ... 204
4.10 Route internal tags (tag-sets in routing policies) ... 205
4.10.1 Order of precedence when multiple applications write internal tags ... 208
4.10.2 Configuring route internal tags ... 209
4.11 EVPN VXLAN Anycast Multi-Homing ... 212
4.11.1 EVPN VXLAN Anycast Multi-Homing for Layer 2 services ... 213
4.11.1.1 Anycast MH ... 213
4.11.1.2 Egress leaf in Anycast MH mode ... 215
4.11.1.3 Ingress leaf in Anycast MH mode ... 217
4.11.1.4 Comparing underlay failure behavior ... 218
4.11.1.5 Egress leaf node failure behavior in Anycast ES ... 218
4.11.2 EVPN VXLAN Anycast Multi-Homing for Layer 3 services ... 219
4.11.2.1 EVPN VXLAN anycast aliasing for Layer 3 routes with ESI-0 ... 220
4.11.2.2 EVPN VXLAN anycast aliasing for Layer 3 routes with non-zero ESI ... 222

5 IP-VPN services ... 226
5.1 Configuring IP-VPN services ... 226

6 BGP VPN Services ... 230
6.1 BGP fast reroute in IP-VRF network instances ... 230
6.2 Inter-domain VPN services ... 230
6.2.1 Multi-instance IP-VRF ... 231
6.2.2 Multi-instance EVPN-IFL (MPLS-VXLAN) ... 233
6.2.3 Multi-instance MAC-VRF ... 235
6.2.4 Inter-domain EVPN IFL and IP-VPN in a single IP-VRF BGP instance ... 243
6.2.5 Domain path (D-PATH) attribute for inter-domain connectivity ... 245
6.2.5.1 D-PATH configuration ... 246
6.2.5.2 D-PATH and loop detection ... 247
6.2.5.3 D-PATH and best path selection ... 248

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

6.2.6 Next-hop-self route reflector and inter-AS option B....248
6.2.6.1 Displaying next-hop-self route reflector and inter-AS option B information....257
6.3 Combined ECMP for BGP PE-CE and EVPN IFL/IP-VPN....260
6.3.1 Combined ECMP example....262
6.4 Micro-segmentation....266
6.4.1 Micro-segmentation topologies....268
6.4.2 Group-based policies....270
6.4.3 Associating groups with interfaces....272
6.4.4 GBP tagging for BGP routes....273
6.4.5 Static routes and micro-segmentation....274
6.4.6 Routing policies and micro-segmentation....276
6.4.7 LPM source lookup mode....277
6.4.8 Group-based policies in VPN services....278
6.4.8.1 Configuration and process....278
6.4.8.2 GBP in an EVPN distributed IRB network....283
6.4.8.3 GBP in an EVPN broadcast domain or centralized IRB network....284
6.4.8.4 GBP configuration and state examples....285
6.5 EVPN ARP/ND synchronization on Layer 3 interfaces....302

7 EVPN configuration examples....309
7.1 All-active redundant connectivity example....309
7.1.1 Configuration for all-active connectivity example....310
7.2 Hierarchical active-active connectivity example....313
7.2.1 Configuration for hierarchical active-active connectivity example....315
7.3 EVPN multihoming as standalone solution for MC-LAG....321
7.3.1 Configuration for EVPN multihoming as standalone MC-LAG....323

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

1 About this guide

This document describes basic configuration for Layer 2 and Layer 3 Virtual Private Network service functionality on the Nokia Service Router Linux (SR Linux). It presents examples to configure and implement various protocols and services.

This document is intended for network technicians, administrators, operators, service providers, and others who need to understand how the router is configured.

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.

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

- Square brackets ([ ]) indicate optional elements.
- 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
Support for gateway-IP load balancing on 7250 IXR Gen 3 platforms	Gateway IP-based load balancing for EVPN IFL routes
Support for inter-domain EVPN IFL/IP-VPN on 7250 IXR Gen 3 platforms	Inter-domain EVPN IFL and IP-VPN in a single IP-VRF BGP instance
Support for Layer 3 proxy-ARP/ND on 7250 IXR Gen 3, 7220 IXR-H5, and 7220 IXR-H6 platforms	Layer 3 proxy-ARP/ND
Support for split horizon groups on MAC-VRF on 7220 IXR-H6 platforms	Split horizon groups on MAC-VRF
Support for EVPN-VXLAN IFL on IP-VRF on 7220 IXR-H6 platforms	Asymmetric IRB Symmetric IRB interface-less ip-vrf-to-ip-vrf model Symmetric IRB model using EVPN MAC/IP advertisement routes in the route table
Support for Layer 2 proxy-ARP/ND on 7220 IXR-H5 and 7220 IXR-H6 platforms	Layer 2 proxy-ARP/ND
Support for unequal ECMP, resilient hashing, and EVPN IP aliasing on 7220 IXR-H5 and 7220 IXR-H6 platforms	Unequal ECMP for EVPN IP prefix routes Resilient hashing for EVPN IFL EVPN IP aliasing
FAT label and Layer 2 attribute signaling for EVPN in MAC-VRF	Flow-aware transport label and Layer 2 attribute signaling for EVPN in MAC-VRF
Best path selection for EVPN AD per-EVI routes	Best path selection for EVPN AD per-EVI routes
EVPN VXLAN anycast multi-homing for Layer 3 services	EVPN VXLAN Anycast Multi-Homing for Layer 3 services
EVPN proxy MAC/IP advertisement routes in MAC-VRFs with IRB interfaces and with proxy-ARP/ND	EVPN Proxy MAC/IP Advertisement routes in MAC-VRFs with IRB interfaces

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

Topic	Location
	EVPN Proxy MAC/IP Advertisement routes in MAC-VRFs with proxy-ARP/ND
PEG function with ingress replication	PEG function with ingress replication

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

3 Overview

This chapter contains overview information about EVPN service infrastructure, EVPN for Layer 2, EVPN for Layer 3, EVPN for multicast, and IP-VPN services.

3.1 About EVPN

Ethernet Virtual Private Network (EVPN) is a technology that bridges Layer 2 and routes Layer 3 VPN traffic across a shared provider environment. When extending Broadcast Domains (BDs), EVPN instances configured on Provider Edge (PE) routers act as virtual bridges, facilitating traffic between Customer Edge (CE) devices at different locations. Conversely, when connecting devices in different IP subnets, EVPN instances function as virtual routers, enabling inter-subnet forwarding for devices of the same tenant.

Essentially, PE routers exchange reachability information, encapsulate Layer 2 or Layer 3 traffic from CE devices, and forward it across the provider or data center (DC) network. EVPN is a standard technology in multi-tenant DCs (RFC 8365) and MPLS/Segment Routing networks (RFC 7432).

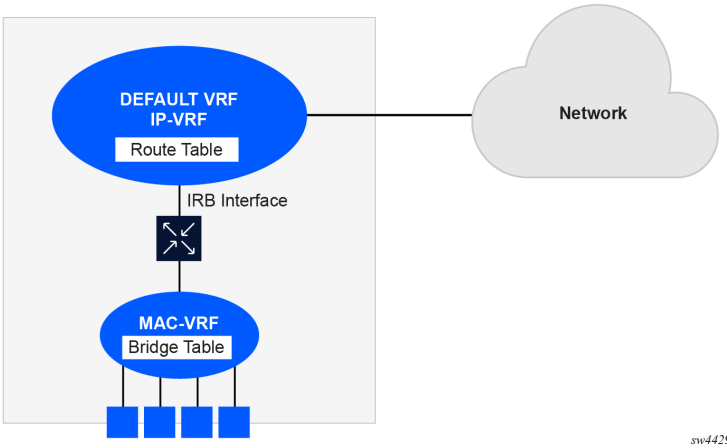
The SR Linux EVPN solution supports EVPN multi-tenant DCs using VXLAN as the data plane, and in service provider networks, using MPLS as the data plane.

3.2 About the service infrastructure for EVPN

The SR Linux infrastructure to support EVPN services uses network-instances of type MAC-VRF and IP-VRF, along with IRB interfaces. The MAC-VRF network-instance is associated with a network-instance of type `default` or `ip-vrf` via an Integrated Routing and Bridging (IRB) interface.

The following figure shows the relationship between an IRB interface and MAC-VRF, and IP-VRF network-instance types.

Figure 1: MAC-VRF, IRB interface, and IP-VRF



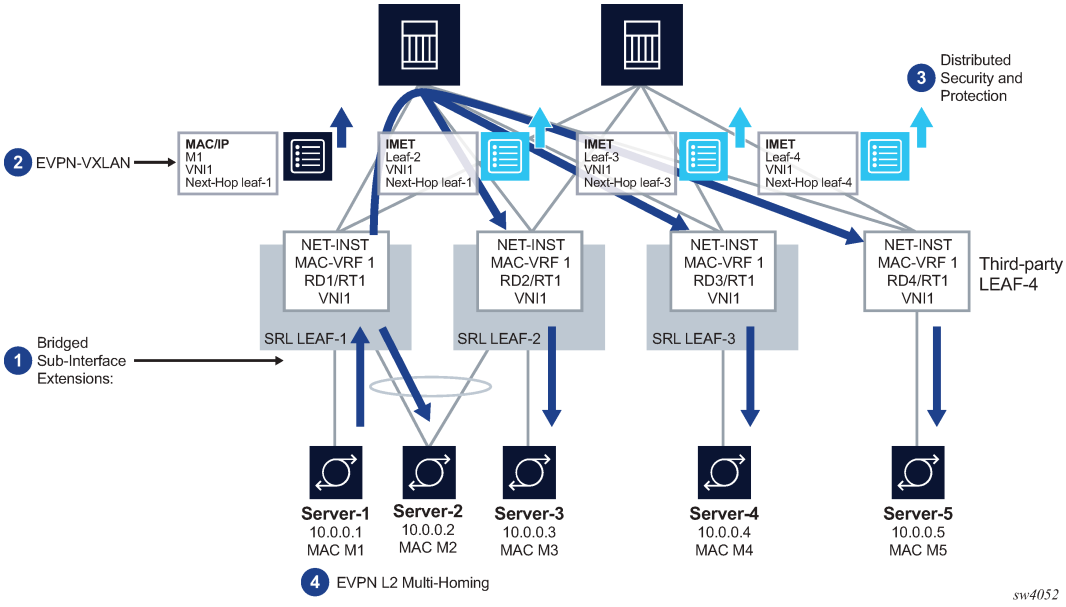
© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

See Layer 2 services infrastructure for information about MAC-VRFs, IP-VRFs, and IRB interfaces in SR Linux.

3.3 About EVPN for Layer 2

The primary usage for EVPN for Layer 2 is the extension of a BD in overlay multi-tenant DCs or service provider networks. An example of this type of topology in DCs is shown in the following figure:

Figure 2: BD extension in overlay DCs



SR Linux features that support this topology fall into the following categories:

- bridged subinterfaces, including:
  - default subinterfaces, configured with the parameter **vlan-id optional**, which capture untagged and non-explicitly configured VLAN-tagged frames on tagged subinterfaces
  - transparency of inner qtags not being used for service classification
- EVPN control and data plane extensions, as described in RFC 8365 for VXLAN tunnels and in RFC 7432 for MPLS tunnels:
  - EVPN routes type MAC/IP and IMET (Inclusive Multicast Ethernet Tag) routes
  - VXLANv4 model for MAC-VRFs
  - MPLS model for MAC-VRFs
- distributed security and protection, including:
  - an extension to the MAC duplication mechanism that can be applied to MACs received from EVPN
  - protection of static MACs
- EVPN Layer 2 multihoming, including:
  - the Ethernet Segment (ES) model definition for all-active and single-active multihoming

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

– interface-level reload-delay timers to avoid service impact when links recover

– load-balancing and redundancy using aliasing, as in RFC 7432

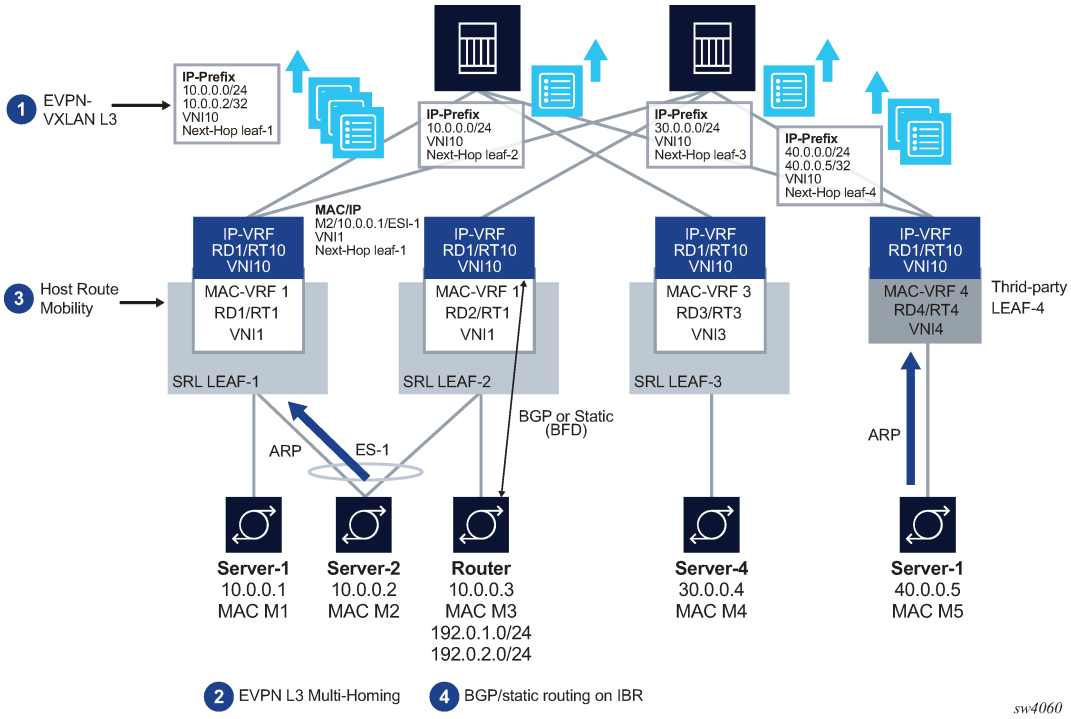
See EVPN for Layer 2 ELAN services for information about the components of EVPN Layer 2 in SR Linux.

3.4 About EVPN for Layer 3

The primary usage for EVPN for VXLAN tunnels (Layer 3) is inter-subnet-forwarding for unicast traffic within the same tenant infrastructure. This type of topology is shown in the following figure:

The primary usage for EVPN for Layer 3 is inter-subnet-forwarding for unicast traffic within the same tenant infrastructure, which can use a shared provider network or a data center. An example of this type of topology in a data center is shown in the following figure:

Figure 3: Inter-subnet forwarding with EVPN Layer 3



SR Linux features that support this topology fall into the following categories:

- EVPN Layer 3 control plane (mostly EVPN IP prefix routes or RT5s) and data plane, as described in RFC 9136
- EVPN Layer 3 multihoming on MAC-VRFs with IRB interfaces that use anycast gateway IP and MAC addresses in all leafs attached to the same BD
- host route mobility procedures to allow fast mobility of hosts between leaf nodes attached to the same BD

Other supported features include:

- interface-less (IFL) model interoperability with unnumbered interface-ful (IFF) model

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

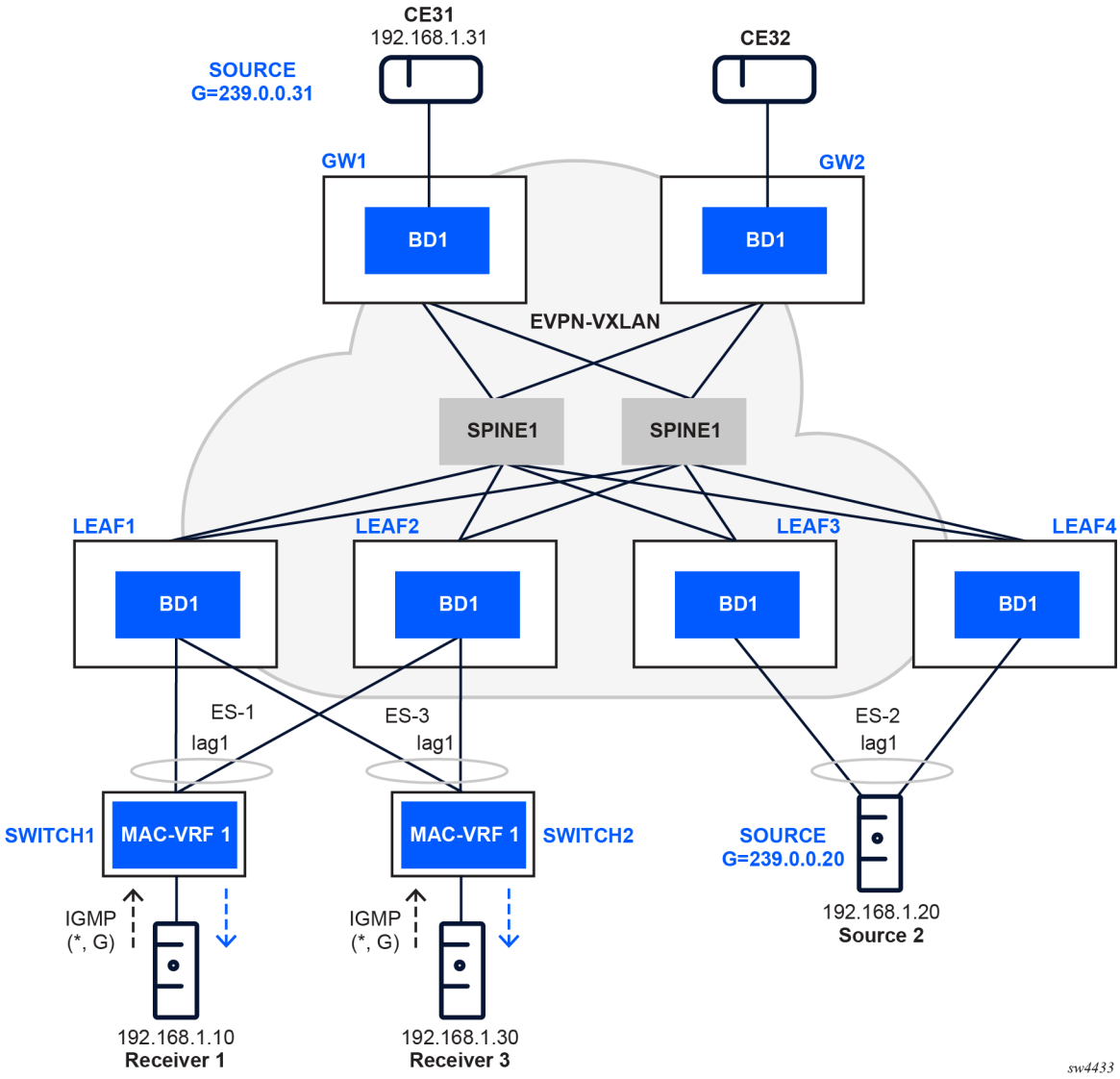
- ECMP over EVPN, including unequal ECMP, IP aliasing, and combined ECMP
- support for interface-level OAM (ping) in anycast deployments

EVPN for Layer 3 describes the components of EVPN Layer 3 in SR Linux.

3.5 About EVPN for multicast

The primary usage for EVPN for multicast is to forward IP multicast traffic from sources to receivers in an efficient way. An example of this type of service in data centers is shown in the following figure:

Figure 4: EVPN for multicast



SR Linux features that support this topology fall into the following categories:

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

- IGMP/MLD snooping on MAC-VRFs
- EVPN IGMP/MLD proxy support in MAC-VRFs, as described in RFC 9251
- EVPN multihoming support on MAC-VRFs where IGMP/MLD snooping is enabled, and synchronization of the multicast states on all the PEs attached to the same Ethernet Segment, as described in RFC 9251

SR Linux also supports inter-subnet multicast forwarding, as specified in RFC9625 (Optimized Inter-Subnet Multicast), where sources and receivers can be located in different subnets.

EVPN for multicast describes the components of EVPN multicast in SR Linux.

3.6 About IP-VPN

IP-VPN services use a combination of MP-BGP and MPLS to distribute IPv4/v6 routing information and provide Layer 3 VPN services.

Each IP-VPN consists of a set of customer end points connected to one or more PE routers. Each associated PE router maintains a separate IP forwarding table for each IP-VPN instance. The PE routers exchange the routing information configured or learned from all customer sites via MP-BGP peering. Each route exchanged via the MP-BGP protocol includes a Route Distinguisher (RD), which identifies the IP-VPN association and handles any potential IP address overlap.

Multi-Protocol BGP (MP-BGP) is used to exchange the routes of a particular VPN among the PE routers that are attached to that VPN.

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

4 EVPN services

This chapter describes the components of various EVPN services on SR Linux.

4.1 Layer 2 services infrastructure

This chapter describes the components of Layer 2 services on SR Linux.

4.1.1 MAC-VRF network instance

The network-instance type MAC-VRF functions as a broadcast domain. Each MAC-VRF network instance builds a bridge table composed of MAC addresses that can be learned via the data path on network instance interfaces or via static configuration. You can configure the size of the bridge table for each MAC-VRF network instance, as well as the aging for dynamically learned MAC addresses and other parameters related to the bridge table.

The MAC-VRF network-instance type features a MAC duplication mechanism that monitors MAC address moves across network-instance interfaces and across interfaces.

4.1.1.1 MAC selection

Each MAC-VRF network instance builds a bridge table to forward Layer 2 frames based on a MAC address lookup. SR Linux selects the MAC addresses to be sent for installation to the line card (XDP), based on the following priority:

1. local application MACs (for example, IRB interface MACs)
2. local static MACs
3. EVPN static MACs
4. local duplicate MACs
5. learned and EVPN-learned MACs

4.1.1.2 MAC duplication detection and actions

MAC duplication is the mechanism used by SR Linux for loop prevention. MAC duplication monitors MAC addresses that move between subinterfaces. It consists of detection, actions, and process restart.

4.1.1.2.1 MAC duplication detection

Detection of duplicate MAC addresses is necessary when extending broadcast domains to multiple leaf nodes. SR Linux supports a MAC duplication mechanism that monitors MAC address moves across network-instance interfaces and across interfaces and EVPN.

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

A MAC address is considered a *duplicate* when its number of detected moves is greater than a configured threshold within a configured time frame where the moves are observed. When the threshold is exceeded, the system holds on to the prior local destination of the MAC and executes an action.

4.1.1.2.2 MAC duplication actions

The action taken on detecting one or more MAC addresses as duplicate on a subinterface can be configured for the MAC-VRF network instance or for the subinterface. The following are the configurable actions:

- **oper-down**

  When one or more duplicate MAC addresses are detected on the subinterface, the subinterface is brought operationally down.

- **blackhole**

  On detection of a duplicate MAC on the subinterface, the MAC is blackholed.

- **stop learning**

  On detection of a duplicate MAC on the subinterface, the MAC address is no longer relearned on this or any subinterface. This is the default action for a MAC-VRF network instance.

- **use-network-instance-action**

  (Only available for subinterfaces) Use the action specified for the MAC-VRF network instance. This is the default action for a subinterface.

4.1.1.2.3 MAC duplication process restarts

When at least one duplicate MAC address is detected, the duplicate MAC addresses are visible in the state datastore and can be displayed with the **info from state mac-duplication duplicate-entries** CLI command. See Displaying bridge table information.

4.1.1.2.4 Configurable hold-down time

The **info from state mac-duplication duplicate-entries** command also displays the hold-down time for each duplicate MAC address. After the hold-down time expires for all of the duplicate MAC addresses for the subinterface, the oper-down or stop-learning action is cleared, and the subinterface is brought operationally up or starts learning again.

The hold-down time is configurable, ranging from 2 and 60 minutes. You can optionally specify **indefinite** for the hold-down time, which prevents the oper-down or stop-learning action from being cleared after a duplicate MAC address is detected; in this case, you can manually clear the oper-down or stop-learning action by changing the **mac-duplication** configuration or using the **tools network-instance bridge-table mac-duplication** command.

4.1.1.3 Bridge table configuration

The bridge table, its MAC address limit, and maximum number of entries can be configured on a per mac-vrf or per-subinterface basis.

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

When the size of the bridge table exceeds its maximum number of entries, the MAC addresses are removed in reverse order of the priority listed in MAC selection.

You can also configure aging for dynamically learned MAC addresses and other parameters related to the bridge table.

4.1.2 Interface extensions for Layer 2 services

To accommodate the Layer 2 services infrastructure, SR Linux interfaces support the following features:

- traffic classification and ingress or egress mapping actions
- subinterfaces of types routed and bridged

4.1.2.1 Traffic classification and ingress/egress mapping actions

On MAC-VRF network instances, traffic can be classified based on VLAN tagging. Interfaces where VLAN tagging is set to **false** or **true** can be used with MAC-VRF network instances.

A default subinterface can be specified, which captures untagged and non-explicitly configured VLAN-tagged frames in tagged subinterfaces.

Within a tagged interface, a default subinterface (where the **vlan-id** value is set to **optional**) and an untagged subinterface can be configured. This type of configuration behaves as follows:

- the **vlan-id optional** subinterface captures untagged and non-explicitly configured VLAN-tagged frames
- the untagged subinterface captures untagged and packets with tag 0 as the outermost tag

When **vlan-id optional** and untagged subinterfaces are configured on the same tagged interface, packets for unconfigured VLANs go to the **vlan-id optional** subinterface and tag 0 or untagged packets go to the untagged subinterface.

Classification is based on the following:

- all traffic, for interfaces where VLAN tagging is set to false, regardless of existing VLAN tags
- single outermost tag for tagged interfaces where VLAN tagging is set to true. By default, only Ethertype 0x8100 is considered for tags; other Ethertypes are treated as payload. The default Ethertype used for classification can be modified by the command **tpid [TPID_0X8100|TPID_0X88A8|TPID_0X9100|TPID_0X9200|TPID_ANY]**.

The following ingress and egress VLAN mapping actions are supported by default:

- at ingress, pop the single outermost tag (tagged interfaces) or perform no user-visible action (untagged interfaces)
- at egress, push a specified tag at the top of the stack (tagged interfaces) or perform no user-visible action (untagged interfaces)
- if the **vlan-id** value is set to **optional** or the subinterface uses an **untagged** configuration, no tag is popped at ingress or pushed at egress

  There is one exception: on a subinterface that uses an **untagged** configuration, if a received packet has tag 0 as its outermost tag, the subinterface pops tag 0.

For more information about subinterface VLAN configuration, Dot1q tagged bridged subinterfaces, and QinQ bridged subinterfaces, see *SR Linux Interfaces Guide*.

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

4.1.2.2 Routed and bridged subinterfaces

SR Linux subinterfaces can be specified as type routed or bridged:

- routed subinterfaces can be assigned to a network instance of type mgmt, default, or ip-vrf
- bridged subinterfaces can be assigned to a network instance of type mac-vrf

Routed subinterfaces allow for configuration of IPv4 and IPv6 settings, and bridged subinterfaces allow for configuration of bridge table and VLAN ingress or egress mapping.

Bridged subinterfaces do not have MTU checks other than the interface-level MTU (port MTU) or the value set with the **l2-mtu** command. The IP MTU is only configurable on routed subinterfaces.

4.1.3 IRB interfaces

IRB interfaces enable inter-subnet forwarding. Network instances of type mac-vrf are associated with a network instance of type ip-vrf via an IRB interface. See Figure 1: MAC-VRF, IRB interface, and IP-VRF for an illustration of the relationship between MAC-VRF and IP-VRF network instances.

On SR Linux, IRB interfaces are named `irb`*N*, where *N* is an integer in the range of 0 to 255. Up to 4095 subinterfaces can be defined under an IRB interface. An IP-VRF network instance can have multiple IRB subinterfaces, while a MAC-VRF network instance can reference only one IRB subinterface.

IRB subinterfaces are type routed. They cannot be configured as type bridged.

IRB subinterfaces operate in the same way as other routed subinterfaces, including support for the following:

- IPv4 and IPv6 access control lists (ACLs)
- static routes and BGP (IPv4 and IPv6 families)
- IP MTU (with the same range of valid values as Ethernet subinterfaces)
- all settings in the subinterface/ipv4 and subinterface/ipv6 containers; for IPv6, the IRB subinterface also gets an IPv6 link local address
- BFD
- subinterface statistics

IRB subinterfaces also support QoS across the following platforms:

Table 3: IRB QoS support on various platforms

Platform	IRB QoS support
7220 IXR-D*x*	Support for DSCP-based QoS (input and output classifiers and rewrite rules).
7250 IXR Gen 2c+ and 7250 IXR Gen 3	No support.
7215 IXS-A1	No support.
7220 IXR-H*x*	Support for DSCP-based QoS (input and output classifiers and rewrite rules).

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

IRB interfaces do not support sFlow, VLAN tagging, or interface statistics.

4.1.3.1 Using ACLs with IRB interfaces and Layer 2 subinterfaces

The following items apply when using access control lists (ACLs) with an IRB interface or Layer 2 subinterface:

- Input ACLs associated with Layer 2 subinterfaces match all the traffic entering the subinterface, including Layer 2 switched traffic or Layer 3 traffic forwarded to the IRB.
- Input ACLs associated with IRB subinterfaces only match Layer 3 traffic, that is, traffic with a MAC destination address (DA) matching the IRB MAC address.
- The same ACL can be attached to a Layer 2 subinterface and an IRB subinterface in the same service. In this case, there are two ACL instances: one for the IRB with higher priority and another one for the bridged traffic. Routed traffic matches the higher priority instance entries of the ACL.
- The same ACL can be attached to IRB subinterfaces and Layer 2 subinterfaces if both subinterface types belong to different services.
- On 7220 IXR-D1, D2, D3, and 7220 IXR-H*x* systems, egress ACLs, unlike ingress ACLs, cannot match both routed and switched traffic when a Layer 2 IP ACL is attached.
- Received traffic on a MAC-VRF is automatically discarded if the MAC source address (SA) matches the IRB MAC address, unless the MAC is an anycast gateway MAC.
- Packet capture filters show the Layer 2 subinterface for switched traffic and the IRB interface for routed traffic.

4.1.4 Layer 2 services configuration

The examples in this section show how to configure a MAC-VRF network instance, bridged interface, and IRB interface.

4.1.4.1 MAC-VRF network instance configuration example

The following example configures a MAC-VRF network instance and settings for the bridge table. The bridge table is set to a maximum of 500 entries. Learned MAC addresses are aged out of the bridge table after 600 seconds.

MAC duplication detection is configured so that a MAC address is considered a duplicate when its number of detected moves across network-instance interfaces is greater than three over a 5-minute interval. In this example, the MAC address is blackholed. After the hold-down time of 3 minutes, the MAC address is flushed from the bridge table, and the monitoring process for the MAC address is restarted.

The example includes configuration for a static MAC address in the bridge table.

The MAC-VRF network instance is associated with a bridged interface and an IRB interface.

```
--{ candidate shared default }--[  ]--
# info with-context network-instance mac-vrf-1
    network-instance mac-vrf-1 {
        type mac-vrf
        admin-state enable
        description "Sample mac-vrf network instance"
        interface ethernet-1/1.1 {
```


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

```
        }
        interface irb1.1 {
        }
        bridge-table {
            mac-learning {
                admin-state enable
                aging {
                    admin-state enable
                    age-time 600
                }
            }
            mac-duplication {
                admin-state enable
                monitoring-window 5
                num-moves 3
                hold-down-time 3
                action blackhole
            }
            mac-limit {
                maximum-entries 500
            }
            static-mac {
                mac 00:00:5E:00:53:FF {
                }
            }
        }
    }
```

4.1.4.2 Bridged subinterface configuration example

The following example configures the bridged subinterface that is associated with the MAC-VRF in the previous example.

```
--{ candidate shared default }--[  ]--
# info with-context interface ethernet-1/1
    interface ethernet-1/1 {
        admin-state enable
        subinterface 1 {
            type bridged
            admin-state enable
            vlan {
                encap {
                    single-tagged {
                        vlan-id 10
                    }
                }
            }
        }
    }
```

The **vlan-id** value can be configured as a specific valid number or with the keyword **optional**, which means any frame that does not hit the **vlan-id** configured in other subinterfaces of the same interface is classified in this subinterface.

In the following example, the **vlan encap untagged** setting is enabled for subinterface 1. This setting allows untagged frames to be captured on tagged interfaces.

For subinterface 2, the **vlan encap single-tagged vlan-id optional** setting allows non-configured VLAN IDs and untagged traffic to be classified to this subinterface.

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

With the **vlan encap untagged** setting on one subinterface, and the **vlan encap single-tagged vlan-id optional** setting on the other subinterface, traffic enters the appropriate subinterface; that is, traffic for unconfigured VLANs goes to subinterface 2, and tag0 or untagged traffic goes to subinterface 1.

```
--{ candidate shared default }--[  ]--
# info with-context interface ethernet-1/2
    interface ethernet-1/2 {
        vlan-tagging true
        subinterface 1 {
            type bridged
            vlan {
                encap {
                    untagged {
                    }
                }
            }
        }
        subinterface 2 {
            type bridged
            vlan {
                encap {
                    single-tagged {
                        vlan-id optional
                    }
                }
            }
        }
    }
```

SR Linux also supports QinQ (or double-tagged) subinterfaces. The following example adds a bridged subinterface that uses inner and outer tags to classify the traffic (in addition to the existing subinterfaces).

Note: QinQ subinterfaces are supported on 7250 IXR platforms.

```
--{ * candidate shared default }--[  ]--
A:srl1# info with-context interface ethernet-1/2
    interface ethernet-1/2 {
        vlan-tagging true
        subinterface 1 {
            type bridged
            vlan {
                encap {
                    untagged {
                    }
                }
            }
        }
        subinterface 2 {
            type bridged
            vlan {
                encap {
                    single-tagged {
                        vlan-id optional
                    }
                }
            }
        }
        subinterface 3 {
            type bridged
            vlan {
```


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

```
                encap {
                    double-tagged {
                        inner-vlan-id 10
                        outer-vlan-id 30
                    }
                }
            }
        }
    }
```

For more information about subinterface VLAN configuration, see the *SR Linux Interfaces Guide*.

4.1.4.3 IRB interface configuration example

The following example configures an IRB interface. The IRB interface is operationally up when its admin-state is enabled, and its IRB subinterfaces are operationally up when associated with MAC-VRF and IP-VRF network instances. At least one IPv4 or IPv6 address must be configured for the IRB subinterface to be operationally up.

```
--{ candidate shared default }--[  ]--
# info with-context interface irb1
    interface irb1 {
        description IRB_Interface
        admin-state enable
        subinterface 1 {
            admin-state enable
            ipv4 {
                admin-state enable
                address 192.168.1.1/24 {
                }
            }
        }
    }
```

4.1.5 Displaying bridge table information

Procedure

You can display information from the bridge table of a MAC-VRF network instance using **show** commands and the **info from state** command.

Example: Display summary information for MAC-VRF network instances

To display a summary of the bridge table contents for the MAC-VRF network instances configured on the system:

```
# show network-instance * bridge-table mac-table summary
-----------------------------------------------------------------
Network-Instance Bridge table summary
-----------------------------------------------------------------
-----------------------------------------------------------------
Name                          : mac-vrf-1
Irb mac                       :   1 Total   1 Active
Static macs                   :  19 Total  19 Active
Duplicate macs                :  10 Total  10 Active
Learnt macs                   :  15 Total  14 Active
Total macs                    :  45 Total  44 Active
```


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

```
Maximum-Entries                 : 200
Warning Threshold Percentage    :  95% (190)
Clear Warning                   :  90% (180)
-------------------------------------------------------------------
Name                            : mac-vrf-2
Irb mac                         :   1 Total   1 Active
Static macs                     :   1 Total   1 Active
Duplicate macs                  :  10 Total  10 Active
Learnt macs                     :  15 Total  14 Active
Total macs                      :  27 Total  26 Active
Maximum-Entries                 : 200
Warning Threshold Percentage    :  95% (190)
Clear Warning                   :  90% (180)
-------------------------------------------------------------------
-------------------------------------------------------------------
Total Irb macs             : 2  Total 2 Active
Total Static macs          : 29 Total 29 Active
Total Duplicate macs       : 20 Total 20 Active
Total Learnt macs          : 30 Total 29 Active
Total Macs                 : 81 Total 80 Active
-------------------------------------------------------------------
```

Example: Display bridge table for a MAC-VRF network instance

To list the contents of the bridge table for a MAC-VRF network instance:

```
# show network-instance mac-vrf-1 bridge-table mac-table all
-----------------------------------------------------------------------------------------
Mac-table of network instance mac-vrf-1
-----------------------------------------------------------------------------------------
+------------------+--------------+----------+---------+------+-----+----------------+
| Mac              |Destination   |Dest-Index|Type     |Active|Aging|Last-update     |
+==================+==============+==========+=========+======+=====+================+
| 00:00:00:00:00:01|ethernet-1/1.1|65        |Learnt   |True  |256  |2020-2-3T3:37:26|
| 00:00:00:00:00:02|        irb1.1| 0        |Irb      |True  |N/A  |2019-2-1T3:37:26|
| 00:00:00:00:00:03|     blackhole| 0        |Duplicate|True  |N/A  |2019-2-1T3:37:26|
| 00:00:00:00:00:04|ethernet-1/1.2|66        |Learnt   |True  |256  |2019-2-1T3:37:26|
-----------------------------------------------------------------------------------------
Total Irb macs             : 2  Total 2 Active
Total Static macs          : 0  Total 0 Active
Total Duplicate macs       : 1  Total 1 Active
Total Learnt macs          : 3  Total 3 Active
Total Macs                 : 6  Total 6 Active
-----------------------------------------------------------------------------------------
```

Example: Display information about a MAC address

To display information about a specific MAC address in the bridge table:

```
# show network-instance * bridge-table mac-table mac 00:00:00:00:00:01
-----------------------------------------------------------------------------------------
Mac-table of network instance mac-vrf-1
-----------------------------------------------------------------------------------------
Mac                     : 00:00:00:00:00:01
Destination             : ethernet-1/1.1
Destination Index       : 65
Type                    : Learnt
Programming status      : Success | Failed | Pending
Aging                   : 250 seconds
Last update             : 2019-12-13T23:37:26.000
Duplicate Detect Time   : N/A
Hold-down-time-remaining: N/A
```


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

```
-------------------------------------------------------------------------------
```

Example: Display duplicate MAC addresses

To display the duplicate MAC address entries in the bridge table:

```
# show network-instance * bridge-table mac-duplication duplicate-entries
-----------------------------------------------------------------------------------------------
Mac-duplication in network instance mac-vrf-1
-----------------------------------------------------------------------------------------------
Admin-state             : Enabled
Monitoring window       : 3 minutes
Number of moves allowed : 5
Hold-down-time          : 10 seconds
Action                  : Stop Learning
-----------------------------------------------------------------------------------------------
Duplicate entries in network instance mac-vrf-1
-----------------------------------------------------------------------------------------------
+------------------+----------------+----------+------------------------+----------------+
| Duplicate mac    | Destination    |Dest-Index| Detect-Time            | Hold down      |
|                  |                |          |                        | time remaining |
+==================+================+==========+========================+================
| 00:00:00:00:00:01|ethernet-1/1.1  |65        |2019-12-13T23:37:26.000 | 6
| 00:00:00:00:00:02|ethernet-1/1.1  |65        |2019-12-13T23:37:26.000 | 6
| 00:00:00:00:00:03|ethernet-1/1.1  |65        |2019-12-13T23:37:26.000 | 6
-----------------------------------------------------------------------------------------------
Total Duplicate macs       : 3  Total 3 Active
-----------------------------------------------------------------------------------------------
```

Example: Use info from state command to display MAC address entries

You can display the duplicate, learned, or static MAC address entries in the bridge table using **info from state** commands. For example, the following command displays the duplicate MAC entries:

```
# info from state network-instance * bridge-table mac-duplication duplicate-entries mac *
 | as table
+------------+------------------+----------+------------------------+---------------+
| Network-   | Duplicate-mac    | Dest-idx | Detect-time            | Hold-down-time|
| instance   |                  |          |                        | remaining     |
+============+==================+==========+========================+===============+
| red        | 00:00:00:00:00:01 |       1  | 2019-12-13T23:37:26.000|            10 |
| red        | 00:00:00:00:00:02 |       2  | 2019-12-13T23:37:26.000|            20 |
| red        | 00:00:00:00:00:03 |       3  | 2019-12-13T23:37:26.000|            90 |
| blue       | 00:00:00:00:00:04 |       4  | 2019-12-13T23:37:26.000|            90 |
| blue       | 00:00:00:00:00:05 |       0  | 2019-12-13T23:37:26.000|            90 |
+------------+------------------+----------+------------------------+---------------+
```

Example: Display learned MAC addresses

The following command displays the learned MAC entries in the table:

```
# info from state network-instance * bridge-table mac-learning learnt-entries mac * | as
 table
+------------------+-----------------+-------------+---------+------------------------+
| Network-instance | Learnt-mac      | Dest        | Aging   | Last-update            |
+==================+=================+=============+=========+========================+
| red              | 00:00:00:00:00:01|ethernet-1/1.1| 300     | 2019-12-13T23:37:26.000|
| red              | 00:00:00:00:00:02|ethernet-1/1.1| 212     | 2019-12-13T23:37:26.000|
| red              | 00:00:00:00:00:03|ethernet-1/1.2| 10      | 2019-12-13T23:37:26.000|
| blue             | 00:00:00:00:00:04|ethernet-1/1.3| 10      | 2019-12-13T23:37:26.000|
| blue             | 00:00:00:00:00:05|ethernet-1/1.4| 20      | 2019-12-13T23:37:26.000|
```


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

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

Example: Display static MAC entries

The following command displays the static MAC entries in the table:

```
# info from state network-instance * bridge-table static-mac mac * | as table
+------------------+------------------+-------------+
| Network-instance | Static-mac       | Dest        |
+==================+==================+=============+
| red              | 00:00:00:00:00:01|ethernet-1/1.1|
| red              | 00:00:00:00:00:02|       irb1.1|
| red              | 00:00:00:00:00:03|    blackhole|
| blue             | 00:00:00:00:00:04|ethernet-1/1.3|
| blue             | 00:00:00:00:00:05|ethernet-1/1.4|
+------------------+------------------+-------------+
```

4.1.6 Deleting entries from the bridge table

Procedure

SR Linux includes commands for deleting duplicate or learned MAC entries from the bridge table. For a MAC-VRF or subinterface, you can delete all MAC entries, MAC entries with a blackhole destination, or a specific MAC entry.

Example: Clear blackhole MAC entries

The following example clears MAC entries in the bridge table for a MAC-VRF network instance that have a blackhole destination:

```
--{ candidate shared default }--[  ]--
# tools network-instance mac-vrf-1 bridge-table mac-duplication delete-blackhole-macs
```

Example: Delete a learned MAC address

The following example deletes a specified learned MAC address from the bridge table for a MAC-VRF network instance:

```
--{ candidate shared default }--[  ]--
# tools network-instance mac-vrf-1 bridge-table mac-learning delete-mac 00:00:00:00:00:04
```

Example: Clear duplicate MAC entries

The following example clears all duplicate MAC entries in the bridge table for a subinterface:

```
--{ candidate shared default }--[  ]--
# tools interface ethernet-1/1.1 bridge-table mac-duplication delete-all-macs
```

4.1.7 Storm control on bridged subinterfaces

Note: Storm control is supported on 7220 IXR-D1/D2/D3/D4/D5 and 7215 IXS-A1 platforms.

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

Storm control allows you to rate limit broadcast, unknown unicast, and multicast (BUM) traffic on bridged subinterfaces. This feature measures the rate for the individual types of BUM traffic, and performs rate limiting based on thresholds configured for each traffic type.

Storm control is configured as an interface-level policer. You set thresholds for each type of BUM traffic. You can configure the threshold as a percentage of interface bandwidth or as a rate in kb/s.

If `kbps` is specified as the unit type, the rate can be configured as a multiple of 64; for example, 64, 128, 192, and so on. If the rate is configured as any value in the 1 to 127 kb/s range, the effective rate is 64 kb/s, which is shown as the operational-rate in **info from state** output. Similarly, if the rate is configured as any value in the 128 to 191 kb/s range, the operational rate is 128 kb/s, and so on.

The default threshold is 100% of the interface bandwidth for the three traffic types. Specifying a threshold of 0 for a traffic type blocks all traffic of that type on the interface. When a storm control policer is set to 0, the first packet of a flow affected by the policer is forwarded, but subsequent packets are dropped.

Storm control policers can be configured for Ethernet interfaces only. The Ethernet interfaces can be members of a LAG, but storm control must be configured on each interface individually; aggregate rate limiting for the LAG is not supported.

Statistics are not collected for the storm control policers. However, it is possible to check discarded packets at the subinterface level. For example, if storm control for broadcast traffic is configured on interface ethernet-1/1 at a rate of 50%, and broadcast traffic on the ethernet-1/1.1 subinterface exceeds 50% of the port capacity, statistics for the ethernet-1/1.1 subinterface show ingress drops.

Storm control on 7220 IXR-D1/D2/D3/D4/D5 platforms

On 7220 IXR-D1/D2/D3/D4/D5 platforms, storm control supports an action for interfaces when the traffic exceeds any of the configured policers. This action is configured by the command **rising-threshold-action** and the possible actions are:

- **disable-interface**

  This action brings down the interface if traffic exceeds the rate in any of the policers. The interface `oper-down-reason` shows `storm-control-action`. To bring the interface back up, the **admin-state** must be toggled (first disable, then enable).

- **trigger-event**

  This action creates an event when a configured rate is exceeded. The event indicates that at least one of the configured storm control rates has been exceeded, but it does not indicate which one. The event is triggered only when the rate rises above the threshold, not when it falls below it. Additionally, if the traffic rate fluctuates rapidly across the threshold, the router may not generate a new event each time.

- **none**

  This is the default action. The 7220 IXR-D*x* platforms allows limited traffic bursts on their storm control policers. These storm control bursts may delay the configured action when the threshold is exceeded and could be incorrectly perceived as a malfunction. Currently, a fixed burst size of 8388 kbits is used on all 7220 IXR-D*x* platforms, and it does not depend on the configured rate.

Storm control on 7215 IXS-A1 platforms

The implementation of storm control on 7215 IXS-A1 platforms differs compared to other platforms.

The configured storm control rates are Layer 1 rates, as opposed to Layer 2 rates. The configured storm control rates include frame preamble and inter-frame gap (IFG) bytes.

For example, on the 7215 IXS-A1, 64-byte frames and a broadcast rate set to 1 000 000 kb/s corresponds to 761 904 kb/s of actual Layer 2 rate: (1 000 000 * 64) / (64+20) = 761 904 kb/s.

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

If storm control is enabled, the `in-discarded-packets` statistics on the interface are increased when packets are dropped because of the storm control policer. However, the subinterface statistics are not modified.

The interface storm control policers are never applied to TCP SYN packets whether their destination MAC is broadcast, multicast, or unknown unicast.

4.1.7.1 Configuring storm control for an interface

Procedure

To configure storm control, set thresholds for each type of BUM traffic, either as a percentage of interface bandwidth or as a rate in `kbps`.

Example: Configure storm control for an interface

The following example configures storm control for an interface on 7220 IXR-D1/D2/D3/D4/D5 and 7215 IXS-A1 platforms. The unit type is specified as `kbps`. Individual thresholds are set for each traffic type.

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/1
    interface ethernet-1/1 {
        admin-state enable
        ethernet {
            storm-control {
                units kbps
                broadcast-rate 128
                multicast-rate 192
                unknown-unicast-rate 225
            }
        }
    }
```

Example: Configure storm control with an action

The following example configures storm control with an action on 7220 IXR-D1/D2/D3/D4/D5 platforms. When the amount of broadcast traffic in the interface exceeds 40% of the interface capacity, the interface is disabled.

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/1
    interface ethernet-1/1 {
        admin-state enable
        ethernet {
            storm-control {
                units percentage
                broadcast-rate 40
                rising-threshold-action disable-interface
            }
        }
    }
```


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

4.1.7.2 Displaying storm control information

Procedure

Display the configured rates for an interface using **info from state** and **show interface** commands.

Example: Display configured rates using info from state command

```
--{ * candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/1 ethernet storm-control
    interface ethernet-1/1 {
        ethernet {
            storm-control {
                units kbps
                broadcast-rate 128
                multicast-rate 172
                unknown-unicast-rate 225
                operational broadcast-rate 128
                operational multicast-rate 192
                operational unknown-unicast-rate 192
            }
        }
    }
```

Example: Display configured rates using show interface command

```
--{ * candidate shared default }--[  ]--
# show interface ethernet-1/1 detail
===============================================================================
Interface: ethernet-1/1
-------------------------------------------------------------------------------
  Description     : dut2
  Oper state      : up
  Down reason     : N/A
  Last change     : 9h21m16s ago, 1 flaps since last clear
  Speed           : 100G
  Flow control    : Rx is disabled, Tx is disabled
  Loopback mode   : false
  MTU             : 9232
  VLAN tagging    : true
  Queues          : 8 output queues supported, 0 used since the last clear
  MAC address     : 00:00:5e:00:53:00
  Last stats clear: never
...
-------------------------------------------------------------------------------
Storm control for ethernet-1/1
-------------------------------------------------------------------------------
  Broadcast Rate (kbps)                    : 128
  Multicast Rate (kbps)                    : 172
  Unknown Unicast (kbps)                   : 225
  Operational Broadcast Rate (kbps)        : 128
  Operational Multicast Rate (kbps)        : 192
  Operational Unknown Unicast Rate (kbps)  : 192
```


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

4.1.8 L2CP transparency

Layer 2 Control Protocol (L2CP) transparency refers to the ability to control whether L2CP frames are tunneled across a carrier Ethernet network. Tunneling in this context means injecting the L2CP frames into the regular Layer 2 datapath, as opposed to trapping them to the CPU or discarding them.

SR Linux identifies L2CP frames according to the destination addresses and protocol identifiers listed in the following table.

Note: XSTP protocol is not supported for L2CP on the 7250 IXR platforms.

Table 4: L2CP frame identification and special handling

Layer 2 control protocol	L2CP destination address and protocol identifier (7220 IXR-D*x*)	L2CP destination address and protocol identifier (7250 IXR-6/10/6e/10e/X1b/X3b)	L2CP destination address and protocol identifier (7220 IXR-H4)
LLDP	DA: 01-80-C2-00-00-0E Ethertype: x88CC	DA: 01-80-C2-00-00-0E Ethertype: x88cc	DA: 01-80-C2-00-00-0E Ethertype: x88CC
LACP	DA: 01-80-C2-00-00-02 Ethertype: 0x8809 Subtype: 0x01	DA: 01-80-C2-00-00-02 Ethertype: 0x8809 Subtype: 0x01	DA: 01-80-C2-00-00-02 Ethertype: 0x8809 Subtype: 0x01 (only untagged PDUs, tagged PDUs cannot be identified by the subtype)
xSTP (STP/RSTP/MSTP)	DA: 01-80-C2-00-00-00	—	DA: 01-80-C2-00-00-00
PAUSE	DA: 01-80-C2-00-00-01	DA: 01-80-C2-00-00-01	DA: 01-80-C2-00-00-01
dot1x	DA: 01-80-C2-00-00-03	DA: 01-80-C2-00-00-03	DA: 01-80-C2-00-00-03
PTP	DA: 01-80-C2-00-00-0E Ethertype: x88F7	DA: 01-80-C2-00-00-0E Ethertype: x88F7	DA: 01-80-C2-00-00-0E Ethertype: x88F7

By default, SR Linux handles L2CP frames according to the default transparency action listed in the following table.

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

Table 5: Default L2CP PDU actions for tagged and untagged frames

Layer 2 control protocol	Default transparency action (7220 IXR-D*x*)	Default transparency action (7250 IXR-6/10/6e/ 10e/X1b/X3b)	Default transparency action (7220 IXR-H4)
LLDP	Trap to CPU only if untagged, drop otherwise	Trap to CPU only if untagged, drop otherwise	Trap to CPU only if untagged, drop otherwise
LACP	Trap to CPU only if untagged, drop otherwise	Trap to CPU only if untagged, drop otherwise	Trap to CPU only if untagged, drop otherwise
xSTP (STP/ RSTP/MSTP)	Drop tagged and untagged	—	Drop tagged and untagged
PAUSE	Drop tagged and untagged	Drop tagged and untagged	—
dot1x	Trap to CPU only if untagged, drop otherwise	—	Trap to CPU only if untagged, drop otherwise
PTP	Drop tagged and untagged	Drop tagged and untagged	Drop tagged and untagged
E-LMI	Drop tagged and untagged	Drop tagged and untagged	Drop tagged and untagged
EFM-OAM	Drop tagged and untagged	Drop tagged and untagged	Drop tagged and untagged
ESMC	Drop tagged and untagged	Trap to CPU only if untagged, drop otherwise	Drop tagged and untagged

The L2CP transparency feature allows you to configure SR Linux to tunnel the following types of L2CP frames instead of performing the default transparency action, or you can configure SR Linux to tunnel all L2CP frames.

When tunneling all l2cp frames, the following protocols are tunneled:

- LLDP
- LACP
- xSTP (STP/RSTP/MSTP)
- dot1x
- PTP

The following protocols can be individually enabled for tunneling within the L2CP transparency:

- LLDP

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

- LACP
- xSTP (STP/RSTP/MSTP)
- PTP

When tunneling is enabled, the following take place:

- Tagged and untagged L2CP frames are classified into a bridged subinterface based on their dot1q tag VLAN ID. If the L2CP frames are classified for forwarding into a MAC-VRF, they are flooded based on their MAC DA. If no bridged subinterface matches the incoming untagged or VLAN-ID tagged frames, they are discarded.
- L2CP transparency is supported on Ethernet interfaces only, with the processing and tunneling possible on bridged subinterfaces of a MAC-VRF. The system does not apply L2CP transparency rules to VXLAN tunnel interfaces, however it does apply them to Layer 3 subinterfaces in the default network instance.
- When an L2CP PDU is tunneled, the source MAC is learned. When the PDU is dropped, the source MAC is not learned.

L2CP transparency is supported in configurations where tunneling for LLDP frames is used in the overlay (between PE and CE devices), and LLDP is used in the underlay (between spine and leaf nodes).

L2CP transparency is not supported on interfaces configured in breakout mode. For LAG interfaces, you must configure L2CP transparency on all member interfaces.

4.1.8.1 Configuring L2CP transparency

Procedure

You can enable tunneling for all L2CP protocol types listed in Table 5: Default L2CP PDU actions for tagged and untagged frames or you can enable tunneling individually for the following L2CP protocol types: LLDP, LACP xSTP (STP/RSTP/MSTP), and PTP.

Example: Enable tunneling for all L2CP protocols

```
--{ candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 ethernet l2cp-transparency
    interface ethernet-1/1 {
        ethernet {
            l2cp-transparency {
                tunnel-all-l2cp true
            }
        }
    }
```

When **tunnel-all-l2cp true** is configured, all L2CP frames coming into the interface, identified by MAC DA = 01:80:c2:00:00:0*x* or MAC DA = 01:80:c2:00:00:2*x* where *x* is any hex value, are tunneled. The exception are PAUSE frames which are never tunnelled.

The **tunnel-all-l2cp** command is not supported on interfaces where LLDP or LACP are enabled.

Note:
Dot1x individual tunneling is configured under the `interface.ethernet.dot1x` context.

Example: Disable tunneling for all L2CP protocols

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


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

```
# info with-context interface ethernet-1/1 ethernet l2cp-transparency
    interface ethernet-1/1 {
        ethernet {
            l2cp-transparency {
                tunnel-all-l2cp false
            }
        }
    }
```

When **tunnel-all-l2cp false** is configured, all L2CP frames not specifically enabled for tunneling are discarded.

Example: Enable tunneling for individual protocol types

You can enable tunneling individually for LLDP, LACP xSTP (STP/RSTP/MSTP), and PTP L2CP protocol types. These L2CP frames are identified by the destination MAC address, Ethertype and sometimes slow-protocol subtype; see Table 5: Default L2CP PDU actions for tagged and untagged frames.

Note: On 7220 IXR-D*x* platforms, the dot1x individual tunneling is configured under the `interface.ethernet.dot1x` context.

The following example enables tunneling for LLDP frames:

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 ethernet l2cp-transparency
    interface ethernet-1/1 {
        ethernet {
            l2cp-transparency {
                lldp {
                    tunnel true
                }
            }
        }
    }
```

By default, all the **tunnel** commands are set to **false**, so all L2CP protocols are discarded, except for LLDP and LACP, for which a copy is trapped to the CPU (if untagged) and another copy is discarded.

Note: Only LLDP frames with MAC 01-80-c2-00-00-0e are tunneled. LLDP frames with MAC DA = 01-80-c2-00-00-00 or 01-80-c2-00-00-03 are not tunneled or trapped to the CPU.

Tunneling for LLDP frames is not supported on an interface where LLDP is enabled.

Tunneling for LACP frames is not supported on an interface where `aggregate-id` is configured or the interface is a member of a LAG where LACP is enabled.

Note:
Dot1x individual tunneling is configured under the `interface.ethernet.dot1x` context.

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

4.1.8.2 Configuring dot1x individual tunneling

Procedure

Note: On 7220 IXR-D*x* platforms, the dot1x individual tunneling is configured under the `interface.ethernet.dot1x` context.

Example

The following example enables dot1x individual tunneling:

```
# info from state with-context interface ethernet-1/31 ethernet dot1x tunnel
    interface ethernet-1/31 {
        ethernet {
            dot1x {
                tunnel {
                    tunnel-all false
                    oper-rule trap-to-cpu-untagged
                }
            }
        }
    }
```

4.1.8.3 Displaying L2CP transparency information

Procedure

Each of the L2CP protocols that can be tunneled is associated with an L2CP transparency rule (`oper-rule`) that indicates the state of the actual rule installed in the datapath. This rule can be any of the following:

- `trap-to-cpu-untagged`
  This rule is used for LLDP, LACP, and PTP frames if they are not configured to be tunneled.

  On 7220 IXR-D*x* and 7250 IXR-6/10/6e/10e/X1b/X3b platforms, untagged frames are trapped to the CPU; that is, a copy is trapped and sent to the CPM, while another copy is discarded. Tagged frames are discarded and not trapped.
- `drop-tagged-and-untagged`
  This rule is used for xSTP, dot1x, and PTP frames if they are not configured for tunneling.
- `tunnel-tagged-and-untagged`
  This rule is used for any of the L2CP protocols when they are configured to be tunneled.

You can display the L2CP transparency configuration and `oper-rule` for each L2CP protocol that can be tunneled.

Example: Display tunneling configuration and oper-rule for an interface on 7220 IXR-D*x* platforms

Use the **info from state** command for an interface to display its tunnelling configuration and `oper-rule` for each protocol. For example:

```
--{ candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/1 ethernet l2cp-transparency
    interface ethernet-1/1 {
```


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

```
        ethernet {
            l2cp-transparency {
                tunnel-all-l2cp false
                lldp {
                    tunnel false
                    oper-rule trap-to-cpu-untagged
                }
                lacp {
                    tunnel false
                    oper-rule trap-to-cpu-untagged
                }
                xstp {
                    tunnel false
                    oper-rule drop-tagged-and-untagged
                }
                ptp {
                    tunnel false
                    oper-rule drop-tagged-and-untagged
                }
            }
        }
    }
```

Example: Display the oper-rule for each protocol type on 7220 IXR-D*x* platforms

Use the **show interface detail** command to display the L2CP transparency rule (`oper-rule`) for each protocol type; for example:

```
--{ candidate shared default }--[  ]--
# show interface ethernet-1/1 detail
===============================================================================
Interface: ethernet-1/1
-------------------------------------------------------------------------------
  Description     : <None>
  Oper state      : up
  Down reason     : N/A
  Last change     : 2d22h27m45s ago, 1 flaps since last clear
  Speed           : 10G
  Flow control    : Rx is disabled, Tx is disabled
  MTU             : 9232
  VLAN tagging    : false
  Queues          : 8 output queues supported, 2 used since the last clear
  MAC address     : 00:01:01:FF:00:01
  Last stats clear: never
  Breakout mode   : false
-------------------------------------------------------------------------------
L2cp transparency rules for ethernet-1/1
-------------------------------------------------------------------------------
  Lldp               : trap-to-cpu-untagged
  Lacp               : trap-to-cpu-untagged
  xStp               : drop-tagged-and-untagged
  Ptp                : tunnel-tagged-and-untagged
  Non-specified L2cp : drop-tagged-and-untagged
-------------------------------------------------------------------------------
```


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

4.1.8.4 Displaying L2CP transparency statistics

Procedure

To display L2CP transparency statistics, use the **info from state** command in candidate or running mode, or the **info** command in state mode.

Statistics are displayed for the following:

- total number of L2CP packets of any L2CP protocol on all interfaces, as well as the total number of discarded and tunneled L2CP packets
- total number of ingress tunneled and trapped-to-CPU packets per L2CP protocol at the system level. Individual statistics are displayed for LLDP, LACP, xSTP, dot1x, and PTP protocols if `tunnel-all-l2cp false` is configured

Example: Display L2CP transparency statistics for 7220 IXR-D*x* platforms

```
--{ candidate shared default }--[  ]--
# info from state with-context system l2cp-transparency l2cp-statistics
    system {
        l2cp-transparency {
            l2cp-statistics {
                total-in-packets 0
                total-in-discarded-packets 0
                total-in-tunneled-packets 0
                total-in-trap-to-cpu-packets 0
                last-clear "a day ago"
                lldp {
                    in-tunneled-packets 0
                    in-trap-to-cpu-packets 0
                }
                lacp {
                    in-tunneled-packets 0
                    in-trap-to-cpu-packets 0
                }
                xstp {
                    in-tunneled-packets 0
                    in-trap-to-cpu-packets 0
                }
                ptp {
                    in-tunneled-packets 0
                    in-trap-to-cpu-packets 0
                }
            }
        }
    }
```

Example: Display L2CP transparency statistics for 7250 IXR-6/10/6e/10e/X1b/X3b platforms

```
--{ + candidate shared default }--[  ]--
# info from state system l2cp-transparency l2cp-statistics
    system {
        l2cp-transparency {
            l2cp-statistics {
                total-in-packets 12
                total-in-discarded-packets 5
                total-in-tunneled-packets 0
                total-in-trap-to-cpu-packets 7
                lldp {
                    in-tunneled-packets 0
```


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

```
                    in-trap-to-cpu-packets 7
                }
                lacp {
                    in-tunneled-packets 0
                    in-trap-to-cpu-packets 0
                }
                ptp {
                    in-tunneled-packets 0
                    in-trap-to-cpu-packets 0
                }
            }
        }
    }
```

Note:

- Discarded L2CP frames are counted as `in-discarded-packets` in interface-level statistics. They are not counted in subinterface statistics.
- Tunneled L2CP frames are counted in interface and subinterface statistics for outgoing interfaces in the appropriate counters.
- If **tunnel-all-l2cp true** is configured, all L2CP frames (including LLDP, LACP, xSTP, dot1x, and PTP) are counted only in `l2cp-statistics/total-in-tunneled-packets` regardless of protocol-specific configuration. In this case, the protocol-specific statistics are displayed as 0.

4.1.8.4.1 Clearing L2CP transparency statistics

Procedure

You can use a **tools** command to clear the total counters and the per-protocol counters for L2CP transparency statistics.

Example: Clear statistics counters for all L2CP protocols

```
--{ running }--[  ]--
# tools system l2cp-transparency l2cp-total-statistics clear
```

Example: Clear statistics counters for a specific L2CP protocol type

```
--{ running }--[  ]--
# tools system l2cp-transparency ptp clear
```

4.1.9 Split horizon groups on MAC-VRF

Note: Split horizon groups are supported on 7220 IXR-D*x*, 7220 IXR-H4, 7220 IXR-H4-32D, 7220 IXR-H5, 7220 IXR-H6, 7250 IXR Gen 2c+, 7250 IXR Gen 3, and 7215 IXS-A1 platforms. See Platform specifications for split horizon groups for more information.

By default, in a MAC-VRF network instance, all EVPN destinations are considered part of the same split horizon group (SHG) and every interface is part of a different SHG. With that default configuration, traffic from a subinterface can be forwarded to other subinterfaces and EVPN destinations of the same MAC-

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

VRF. However, traffic from EVPN destinations can only be forwarded to subinterfaces, but not to other EVPN destinations.

The explicit configuration of SHGs in a MAC-VRF network instance modifies this default behavior. SHGs can be configured and associated with bridged subinterfaces and BGP-EVPN instances.

SHGs are typically used to prevent client-to-client communication while allowing client-to-server communication. See MPLS transport for EVPN services.

With explicit SHGs, traffic (unicast and BUM) cannot be forwarded between objects that are part of the same SHG.

For control plane traffic, incoming traffic on object A cannot be reinjected by the CPM on object B, if both object A and B are configured with the same SHG. An example of this behavior is proxy-ARP and two Layer 2 subinterfaces configured with the same SHG and attached to the same MAC-VRF. An ARP request received on one of the two subinterfaces cannot be reinjected into the other subinterface if the proxy-ARP lookup fails to find a successful entry.

4.1.9.1 Configuring split horizon groups

Procedure

Before you can reference them in any object, you must first manually create SHGs using the following command: **network-instance bridge-table split-horizon-group** *name*

When created, SHGs are only used in the context of the MAC-VRF.

You can reference SHGs in the following contexts:

- network instance interfaces

  network-instance interface bridge-table split-horizon-group *name*
- BGP-EVPN MPLS

  network-instance protocols bgp-evpn bgp-instance mpls bridge-table split-horizon-group *name*
- connection points

  network-instance connection-point bridge-table split-horizon-group *name*

4.1.9.2 Split horizon group configuration example

The following example illustrates the use of SHGs.

Interfaces **ethernet-1/1.2** and **ethernet-1/1.3** cannot forward traffic between each other, but can send traffic to the other interfaces and EVPN destinations.

Interfaces **ethernet-1/2.1** and **ethernet-1/2.2** cannot forward traffic between each other, but can send traffic to the other interfaces and EVPN destinations.

Interface **ethernet-1/2.3** can forward traffic normally to any other interface or EVPN destination.

Example: Configuring split horizon groups

```
--{ * candidate shared default }--[ network-instance * ]--
A:leaf1# info
    network-instance MAC-VRF-2 {
        interface ethernet-1/1.2 {
            bridge-table {
```


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

```
                split-horizon-group SHG-1
            }
        }
        interface ethernet-1/1.3 {
            bridge-table {
                split-horizon-group SHG-1
            }
        }
        interface ethernet-1/2.1 {
            bridge-table {
                split-horizon-group SHG-2
            }
        }
        interface ethernet-1/2.2 {
            bridge-table {
                split-horizon-group SHG-2
            }
        }
        interface ethernet-1/2.3 {
        }
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    encapsulation-type vxlan
                    vxlan-interface vxlan0.1
                    evi 2
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
        bridge-table {
            split-horizon-group SHG-1 {
            }
            split-horizon-group SHG-2 {
            }
        }
    }
```

4.1.9.3 Platform specifications for split horizon groups

The following limitations for SHGs apply depending on the platform.

7215 IXS-A1 platform support

For expected SHG behavior, all bridged subinterfaces in a MAC-VRF are assigned to the SHG. A bridged subinterface in an SHG drops (at egress) all packets coming from another bridged subinterface, whether the origin is a subinterface in the same SHG or not.

SHGs on 7215 IXS-A1 platforms have the following capabilities:

- In MAC-VRFs without IRB, if all subinterfaces are applied to an SHG there is no connectivity among them. If not all subinterfaces are associated with the SHG, traffic from SHG subinterfaces to non-SHG subinterfaces is forwarded, but traffic from non-SHG subinterfaces to SHG subinterfaces is dropped.

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

- In MAC-VRFs with IRB, all subinterfaces of the MAC-VRF must be configured on the SHG. Traffic among subinterfaces is dropped, but traffic to and from IRB is forwarded correctly.

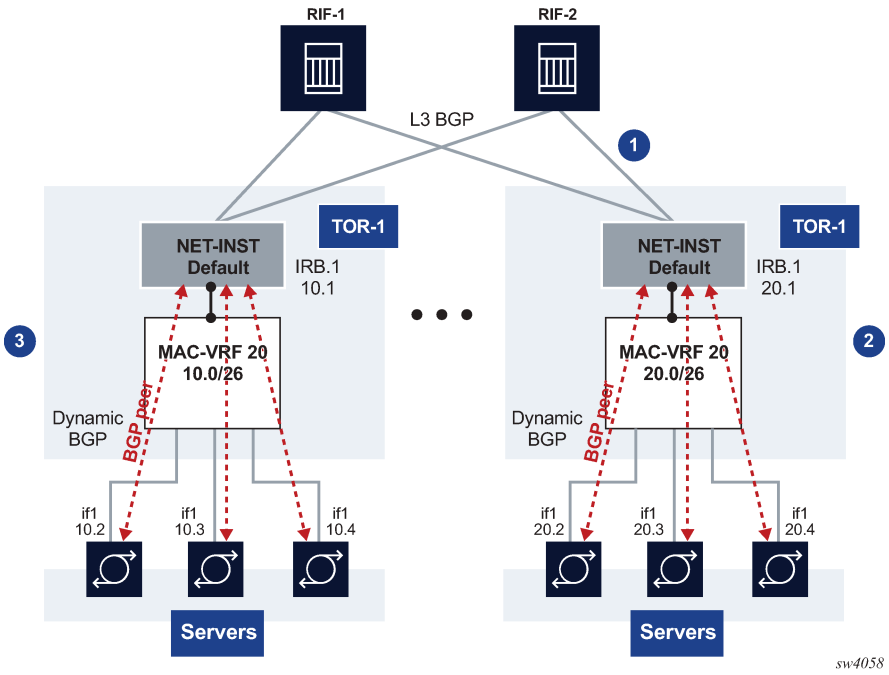
4.1.10 Server aggregation configuration example

Figure 5: Server aggregation example shows an example of using MAC-VRF network instances to aggregate servers into the same subnet.

In this example, Leaf-1 and Leaf-2 are configured with MAC-VRF instances that aggregate a group of servers. These servers are assigned IP addresses in the same subnet and are connected to the leaf default network instance by a single IRB subinterface. The servers use a PE-CE BGP session with the IRB IP address to exchange reachability.

Using the MAC-VRF with an IRB subinterface saves routed subinterfaces on the default network instance; only one routed subinterface is needed, as opposed to one per server.

Figure 5: Server aggregation example



In this example:

1. TORs peer (eBGP) to two or four RIFs.
2. MAC-VRF 20 is defined in TORs with an IRB interface with IPv4/IPv6 addresses. DHCP relay is supported on IRB subinterfaces.
3. The following Layer 2 features are implemented or loop and MAC duplication protection:
   - MAC duplication with oper-down or blackhole actions configured on the bridged subinterfaces
   - storm control for BUM traffic on bridged subinterfaces

This example uses the following features:

- MAC-VRF with bridge subinterfaces and IRB subinterfaces to the default network instance

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

- PE-CE BGP sessions for IPv4 and IPv6 address families
- MAC duplication with oper-down or blackhole actions configured on the bridged subinterfaces
- storm control for BUM traffic on bridged subinterfaces

4.1.10.1 Configuration for server aggregation example

The following shows the configuration of Leaf-1 in Figure 5: Server aggregation example and its BGP session via IRB to server 1. Similar configuration is used for other servers and other TORs.

```
--{ [FACTORY] + candidate shared default }--[  ]--
A:Leaf-1# info with-context interface ethernet-1/1
    interface ethernet-1/1 {
        description tor1-server1
        vlan-tagging true
        subinterface 1 {
            type bridged
            vlan {
                encap {
                    single-tagged {
                        vlan-id 100
                    }
                }
            }
        }
    }

// Configure an IRB interface and sub-interface that will connect
 the MAC-VRF to the existing default network-instance.

--{ [FACTORY] + candidate shared default }--[  ]--
A:Leaf-1# info with-context interface irb1
    interface irb1 {
        subinterface 1 {
            ipv4 {
                admin-state enable
                address 10.0.0.2/24 {
                }
            }
            ipv6 {
                admin-state enable
                address 2001:db8::2/64 {
                }
            }
        }
    }

// Configure the network-instance type mac-vrf
and associate the bridged and irb interfaces to it.

--{ [FACTORY] + candidate shared default }--[  ]--
A:Leaf-1# info with-context network-instance MAC-VRF-1
    network-instance MAC-VRF-1 {
        type mac-vrf
        admin-state enable
        interface ethernet-1/1.1 {
        }
        interface irb1.1 {
        }
    }
```


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

```
// Associate the same IRB interface to the network-
instance default and configure the BGP IPv4 and IPv6 neighbors to DUT1 and DUT3.

--{ [FACTORY] + candidate shared default }--[  ]--
A:Leaf-1# info
    network-instance default {
        type default
        admin-state enable
        router-id 2.2.2.2
        interface irb1.1 {
        }
        protocols {
            bgp {
                admin-state enable
                afi-safi ipv4-unicast {
                    admin-state enable
                }
                autonomous-system 64502
                router-id 10.0.0.2
                ebgp-default-policy {
                    import-reject-all false
                }
                failure-detection {
                    enable-bfd true
                    fast-failover true
                }
                group leaf {
                    admin-state enable
                    export-policy pass-all
                    afi-safi ipv4-unicast {
                        admin-state enable
                    }
                    afi-safi ipv6-unicast {
                        admin-state enable
                    }
                    local-as as-number 64502 {
                    }
                    timers {
                        minimum-advertisement-interval 1
                    }
                }
                afi-safi ipv4-unicast {
                    admin-state enable
                }
                afi-safi ipv6-unicast {
                    admin-state enable
                }
                neighbor 10.0.0.1 {
                    peer-as 64501
                    peer-group leaf
                    transport {
                        local-address 10.0.0.2
                    }
                }
                neighbor 2001:db8::1 {
                    peer-as 64501
                    peer-group leaf
                    transport {
                        local-address 2001:db8::2
                    }
                }
            }
        }
    }
```


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

4.1.11 Spanning Tree Protocol (STP)

Note: This feature is supported on 7220 IXR-D2, 7220 IXR-D2L, 7220 IXR-D3, 7220 IXR-D3L, 7220 IXR-D4, and 7220 IXR-D5 platforms.

In bridged networks, ensuring full connectivity and avoiding network loops is critical. STP provides loop-free topology across networks comprising multiple interconnected bridges or switches. To optimize network performance and resilience, SR Linux devices support two modes of STP, Rapid Spanning Tree Protocol (RSTP) and Multiple Spanning Tree Protocol (MSTP).

RSTP

RSTP, defined in IEEE 802.1w, provides faster convergence and better network stability than legacy STP. It operates on a single spanning tree for each network segment and maintains loop-free connectivity in networks where SR Linux devices are interconnected with computers, personal laptops, unmanaged or managed switches, and Wi-Fi hubs. Broadcast storms originating from these access ports can severely disrupt network operations. Therefore, enabling STP capabilities on all access ports is essential to maintain network stability. RSTP is supported on a per-MAC-VRF and per-subinterface basis. When STP is enabled on an SR Linux device, RSTP is the default mode, offering backward compatibility with legacy STP implementations.

MSTP

MSTP extends the concept of RSTP by grouping and associating VLANs with Multiple Spanning Tree Instances (MSTI). Each MSTI can maintain its own topology enabling load balancing by providing multiple forwarding paths. Network fault tolerance is also improved because a failure in one instance (forwarding path) does not affect other instances.

SR Linux devices support MSTP in a management STP instance that can contain up to 15 MSTIs. The management STP instance is used only for processing STP Bridge Protocol Data Units (BPDUs) that are tagged and untagged. Features such as EVPN, IRB, and bridge tables are not supported in this instance.

For devices to take part in MSTIs, they must all use the same configuration. A group of interconnected devices with identical Multiple Spanning Tree (MST) settings such as region name, revision number, and VLAN-to-instance mapping is called an MST region.

MSTI 0 is a special instance for a region, known as the Common Internal Spanning Tree (CIST) instance. The CIST exists by default, managing all subinterfaces and carrying priority information in untagged BPDUs. Different bridges can operate as the root for different MSTIs by adjusting bridge priorities. MAC-VRFs inherit MSTP states from the associated MSTI or CIST and support advanced features such as EVPN, IRB, and bridge tables.

BPDU

STP relies on messages called BPDUs to build and maintain the network’s spanning tree. BPDUs are sent from one device to another and contain information about the device such as its bridge ID, root path cost, and port MAC addresses. When SR Linux devices first exchange BPDUs, they use this information to select the root bridge. BPDUs also communicate the cost of each network link, which depends on the port speed or user configuration.

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

BPDU guard

BPDU guard is a protection feature in RSTP, and MSTP that prevents accidental connection to edge ports. BPDUs are exchanged by SR Linux devices to run STP and prevent loops. Edge ports where end devices like PCs, printers, or servers connect are not supposed to participate in STP. If such a port starts receiving BPDUs accidentally, it disrupts the STP topology. BPDU guard prevents this scenario by shutting down the port immediately upon receiving a BPDU. You can enable BPDU guard only when the **edge-port** is set to **EDGE_ENABLE**.

You can enable BPDU guard either for each subinterface or globally on MSTP interfaces. All MAC-VRFs with MSTP-managed subinterfaces inherit the BPDU guard protection when enabled on MSTP interface.

BPDU guard recovery time

You can globally configure how long a subinterface remains operationally down after a BPDU guard error is detected. After the timer expires, the BPDU guard error clears automatically and the subinterface reverts to operationally up state. The following considerations apply for BPDU guard recovery time configuration in SR Linux devices:

- If you change the recovery timer value, the new setting takes effect in the next timer cycle. It does not update the currently running timer.
- If you manually disable a subinterface or interface, the BPDU guard error (and its timer, if running) is cleared. The BPDU guard mechanism restarts automatically when the interface is re-enabled.
- If BPDU guard itself is disabled at the configuration level, any existing BPDU guard error and running timer are cleared. The mechanism restarts when BPDU guard is re-enabled.
- If a port is removed (not present) or the system is rebooted, both the BPDU guard error and its timer are cleared. The BPDU guard mechanism restarts after the port becomes active again.

STP configuration considerations

The following considerations apply for STP configuration in SR Linux devices:

- STP cannot be enabled on bridged subinterfaces with dot1q VLAN ranges.
- When RSTP is enabled on a MAC-VRF, or when a MAC-VRF is managed by a management STP instance, avoid configuring multiple subinterfaces under the same interface.
- In an MSTI using MSTP, you must configure unique, non-overlapping VLAN ID ranges for each MSTI.
- STP cannot be enabled on a MAC-VRF if any of its interfaces have Layer-2 control protocol transparency enabled.
- In a MAC-VRF managed by MSTP, each subinterface must belong to only one MSTI. This restriction also applies to MSTI 0.
- In a MAC-VRF with RSTP enabled, only one untagged interface is allowed, either as an untagged interface or an optional interface.
- You cannot enable RSTP on MAC-VRFs that are managed by MSTP.
- VLAN range subinterfaces are not allowed on a management STP interface.
- An interface cannot be part of more than one MSTI.
- During application warm restart, STP states are maintained. However, no topology change events are processed.
- STP and L2CP transparency tunneling cannot be enabled simultaneously on the same interface. If **l2cp-transparency tunnel-all-l2cp xstp tunnel** is enabled on an interface where STP is already

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

configured, the commit fails. On the 7220 IXR-D1, 7220 IXR-D2, and 7220 IXR-D3 platforms, an additional node-level restriction applies. If Mgmt-STP (MSTP or RSTP mode) runs on any interface, xSTP tunneling cannot be enabled on any interface on the node.

4.1.11.1 Configuring STP on a bridged subinterface

About this task

Perform the following steps to configure STP on a bridged subinterface.

Procedure

Step 1. Configure the subinterface type as **bridged** using the **interface subinterface type** command.

Step 2. Configure the STP parameters using the **interface subinterface bridge-table stp** command as shown in the following example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 subinterface 1
    interface ethernet-1/1 {
        subinterface 1 {
            type bridged
            admin-state enable
            bridge-table {
                stp {
                    admin-state enable
                    port-number 1
                    priority 2
                    path-cost 10
                    edge-port EDGE_ENABLE
                    link-type P2P
                    guard ROOT
                    bpdu-guard true
                }
            }
        }
    }
```

Step 3. View the status of STP using the **info from state with-context interface subinterface bridge-table stp** command as shown in the following example.

Example

```
--{ + candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/1 subinterface 1 bridge-table stp
    interface ethernet-1/1 {
        subinterface 1 {
            bridge-table {
                stp {
                    admin-state enable
                    port-number 1
                    priority 2
                    path-cost 10
                    edge-port EDGE_ENABLE
                    link-type P2P
                    guard ROOT
                    bpdu-guard true
                    port-state DISCARDING
```


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

```
                    port-num 1
                    oper-port-priority 0
                    port-role DISABLED
                    oper-state down
                    designated-bridge 0.0.00:00:00:00:00:00
                    designated-port-priority 0
                    designated-port-num 0
                    oper-protocol rstp
                    oper-bpdu-encap dot1d
                    oper-edge EDGE_ENABLE
                    forward-transitions 0
                    bpdu-guard-error false
                    bpdu-guard-recovery-time-expires 0
                    statistics {
                        bad-bpdus-received 0
                        cfg-bpdus-received 0
                        cfg-bpdus-transmitted 0
                        tcn-bpdus-received 0
                        tcn-bpdus-transmitted 0
                        tc-bit-bpdus-received 0
                        tc-bit-bpdus-transmitted 0
                        rst-bpdus-received 0
                        rst-bpdus-transmitted 0
                        mst-bpdus-received 0
                        mst-bpdus-transmitted 0
                    }
                }
            }
        }
    }
```

4.1.11.2 Configuring STP on a MAC-VRF

About this task

Perform the following steps to configure STP on a MAC-VRF network instance.

Procedure

Step 1. Configure the network instance type as **mac-vrf** using the **network-instance type** command.

Step 2. Configure the STP parameters using the **network-instance protocols stp** command as shown in the following example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance stp-test
    network-instance stp-test {
        type mac-vrf
        admin-state enable
        protocols {
            stp {
                admin-state enable
                bridge-priority 32768
                bridge-address 1A:FA:00:FF:00:00
                forward-delay 15
                hello-time 2
                hold-count 6
                max-age 12
                trace-options {
```


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

```
                    trace [
                        all
                    ]
                }
            }
        }
    }
```

Step 3. View the status of STP using the **info from state with-context network-instance** command as shown in the following example.

Example

```
--{ + candidate shared default }--[  ]--
# info from state with-context network-instance stp-test
    network-instance stp-test {
        type mac-vrf
        admin-state enable
        oper-state up
        oper-mac-vrf-mtu 9232
        interface stp1 {
            oper-state up
            index 4
            interface-ref {
                interface ethernet-1/1
                subinterface 1
            }
            bridge-table {
                mac-relearn-only false
                multicast-forwarding BUM
                oper-mac-learning up
            }
        }
        protocols {
            stp {
                admin-state enable
                bridge-priority 32768
                bridge-address 1A:FA:00:FF:00:00
                forward-delay 15
                hello-time 2
                hold-count 6
                max-age 12
                oper-state down
                bridge-id 32768.1.1A:FA:00:FF:00:00
                designated-root 0.0.00:00:00:00:00:00
                root-port 0
                root-cost 0
                topology-changes 0
                topology-change-active false
                time-since-topology-change "2025-08-29T20:01:03.000Z (now)"
                oper-max-age 0
                oper-hello-time 0
                oper-forward-delay 0
                trace-options {
                    trace [
                        all
                    ]
                }
            }
```


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

4.1.11.3 Configuring a management STP instance

About this task

Perform the following steps to configure a management STP instance.

Procedure

Step 1. Configure the management STP instance name using the **system network-instance protocols mgmt-stp** command specifying the name of the management STP instance.

Step 2. Configure the management STP instance parameters as shown in the following example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context system network-instance protocols
    system {
        network-instance {
            protocols {
                mgmt-stp test1 {
                    mode mstp
                    admin-state enable
                    bridge-priority 32768
                    bridge-address 1A:FA:00:FF:00:00
                    forward-delay 15
                    hello-time 2
                    hold-count 6
                    max-age 20
                    mst-instance 1 {
                        mst-priority 32768
                        vlan-range [
                            1..10
                            100..200
                        ]
                    }
                    mst-instance 2 {
                        mst-priority 8192
                        vlan-range [
                            500..555
                        ]
                    }
                    interface test-stp10 {
                        edge-port EDGE_ENABLE
                        guard NONE
                        bpdu-guard true
                        interface-ref {
                            interface ethernet-1/10
                        }
                        mst-instance 1 {
                        }
                        mst-instance 2 {
                        }
                    }
                    interface test-stp12 {
                        admin-state enable
                        edge-port EDGE_DISABLE
                        link-type P2P
                        guard NONE
                        interface-ref {
                            interface ethernet-1/12
                        }
```


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

```
                        mst-instance 1 {
                        }
                        mst-instance 2 {
                        }
                    }
                }
            }
        }
    }
```

Step 3. View the status of management STP instance using the **info from state with-context system network-instance protocols mgmt-stp** command as shown in the following example.

Example

```
--{ + candidate shared default }--[  ]--
# info from state with-context system network-instance protocols mgmt-stp test1
    system {
        network-instance {
            protocols {
                mgmt-stp test1 {
                    mode mstp
                    admin-state enable
                    bridge-priority 32768
                    bridge-address 1A:FA:00:FF:00:00
                    forward-delay 15
                    hello-time 2
                    hold-count 6
                    max-age 20
                    mst-revision 0
                    mst-max-hops 20
                    oper-state up
                    bridge-id 32768.0.1A:FA:00:FF:00:00
                    designated-root 32768.0.1A:FA:00:FF:00:00
                    root-port 0
                    root-cost 0
                    topology-changes 0
                    topology-change-active false
                    time-since-topology-change "2025-08-29T20:36:15.000Z (now)"
                    oper-max-age 20
                    oper-hello-time 2
                    oper-forward-delay 15
                    cist-regional-root 32768.0.1A:FA:00:FF:00:00
                    cist-internal-root-cost 0
                    cist-remaining-hop-count 20
                    cist-regional-root-port 0
                    mst-instance 1 {
                        mst-priority 32768
                        regional-root 32768.1.1A:FA:00:FF:00:00
                        internal-root-cost 0
                        remaining-hop-count 20
                        root-port 0
                        vlan-range [
                            1..10
                            100..200
                        ]
                    }
                    mst-instance 2 {
                        mst-priority 8192
                        regional-root 8192.2.1A:FA:00:FF:00:00
                        internal-root-cost 0
                        remaining-hop-count 20
                        root-port 0
                        vlan-range [
```


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

```
                            500..555
                        ]
                    }
                    interface test-stp10 {
                        admin-state disable
                        priority 128
                        path-cost 10
                        edge-port EDGE_ENABLE
                        link-type P2P
                        guard NONE
                        bpdu-guard true
                        port-state DISCARDING
                        port-num 2048
                        oper-port-priority 128
                        port-role DISABLED
                        oper-state down
                        designated-bridge 0.0.00:00:00:00:00:00
                        designated-port-priority 0
                        designated-port-num 0
                        oper-protocol rstp
                        oper-bpdu-encap dot1d
                        oper-edge EDGE_ENABLE
                        forward-transitions 0
                        bpdu-guard-error false
                        bpdu-guard-recovery-time-expires 0
                        interface-ref {
                            interface ethernet-1/10
                        }
                        statistics {
                            bad-bpdus-received 0
                            cfg-bpdus-received 0
                            cfg-bpdus-transmitted 0
                            tcn-bpdus-received 0
                            tcn-bpdus-transmitted 0
                            tc-bit-bpdus-received 0
                            tc-bit-bpdus-transmitted 0
                            rst-bpdus-received 0
                            rst-bpdus-transmitted 0
                            mst-bpdus-received 0
                            mst-bpdus-transmitted 0
                        }
                        mst-instance 1 {
                            mst-path-cost 10
                            mst-port-priority 128
                            oper-mst-port-priority 128
                            inside-region false
                            port-state DISCARDING
                            port-role DISABLED
                            designated-bridge 0.0.00:00:00:00:00:00
                            designated-port-priority 0
                            designated-port-num 0
                        }
                        mst-instance 2 {
                            mst-path-cost 10
                            mst-port-priority 128
                            oper-mst-port-priority 128
                            inside-region false
                            port-state DISCARDING
                            port-role DISABLED
                            designated-bridge 0.0.00:00:00:00:00:00
                            designated-port-priority 0
                            designated-port-num 0
                        }
                    }
```


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

```
                    interface test-stp12 {
                        admin-state enable
                        priority 128
                        path-cost 10
                        edge-port EDGE_DISABLE
                        link-type P2P
                        guard NONE
                        bpdu-guard false
                        port-state DISCARDING
                        port-num 2049
                        oper-port-priority 128
                        port-role DISABLED
                        oper-state down
                        designated-bridge 0.0.00:00:00:00:00:00
                        designated-port-priority 0
                        designated-port-num 0
                        oper-protocol mstp
                        oper-bpdu-encap dot1d
                        oper-edge EDGE_DISABLE
                        forward-transitions 0
                        bpdu-guard-error false
                        bpdu-guard-recovery-time-expires 0
                        interface-ref {
                            interface ethernet-1/12
                        }
                        statistics {
                            bad-bpdus-received 0
                            cfg-bpdus-received 0
                            cfg-bpdus-transmitted 0
                            tcn-bpdus-received 0
                            tcn-bpdus-transmitted 0
                            tc-bit-bpdus-received 0
                            tc-bit-bpdus-transmitted 0
                            rst-bpdus-received 0
                            rst-bpdus-transmitted 0
                            mst-bpdus-received 0
                            mst-bpdus-transmitted 0
                        }
                        mst-instance 1 {
                            mst-path-cost 10
                            mst-port-priority 128
                            oper-mst-port-priority 128
                            inside-region false
                            port-state DISCARDING
                            port-role DISABLED
                            designated-bridge 0.0.00:00:00:00:00:00
                            designated-port-priority 0
                            designated-port-num 0
                        }
                        mst-instance 2 {
                            mst-path-cost 10
                            mst-port-priority 128
                            oper-mst-port-priority 128
                            inside-region false
                            port-state DISCARDING
                            port-role DISABLED
                            designated-bridge 0.0.00:00:00:00:00:00
                            designated-port-priority 0
                            designated-port-num 0
                        }
                    }
                }
            }
        }
```


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

```
    }
```

4.1.11.4 Configuring RSTP mode on a management STP instance

Procedure

To set the RSTP mode on management STP instance, use the **system network-instance protocols mgmt-stp mode** command specifying the mode as **rstp**.

Example: Configuring the STP mode

The following example configures the RSTP mode.

```
--{ + candidate shared default }--[  ]--
# info with-context system network-instance protocols mgmt-stp test-stp
    system {
        network-instance {
            protocols {
                mgmt-stp test-stp {
                    mode rstp
                }
            }
        }
    }
```

4.1.11.5 Configuring BPDU guard

Procedure

To enable the BPDU guard on a subinterface, use the **interface subinterface bridge-table stp edge-port** command to configure the edge port as **EDGE_ENABLE** and configure the **bpdu-guard** parameter as **true**. See Configuring BPDU guard on a subinterface

.
To enable the BPDU guard on a management STP interface, use the **system network-instance protocols mgmt-stp interface edge-port** command to configure the edge port as **EDGE_ENABLE** and configure the **bpdu-guard** parameter as **true**. In this case, all MAC-VRFs with MSTP-managed subinterfaces inherit the BPDU guard protection. See Configuring BPDU guard on a management STP interface.

Example: Configuring BPDU guard on a subinterface

The following example configures the BPDU guard on a subinterface.

```
--{ + candidate shared default }--[  ]--
# info with-context interface ethernet-1/10 subinterface 1 bridge-table stp
    interface ethernet-1/10 {
        subinterface 1 {
            bridge-table {
                stp {
                    admin-state enable
                    edge-port EDGE_ENABLE
                    bpdu-guard true
                }
            }
```


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

```
        }
    }
```

Example: Configuring BPDU guard on a management STP interface

The following example configures the BPDU guard on a management STP interface

```
--{ + candidate shared default }--[  ]--
# info with-context system network-instance protocols mgmt-stp 1 interface 1
    system {
        network-instance {
            protocols {
                mgmt-stp 1 {
                    interface 1 {
                        edge-port EDGE_ENABLE
                        bpdu-guard true
                    }
                }
            }
        }
    }
```

4.1.11.6 Configuring BPDU guard recovery time

Procedure

To configure the BPDU guard recovery time, use the **system protocols stp bpdu-guard-recovery-time** command and specify how long, in minutes, a subinterface can remain operationally down after a BPDU guard error is detected.

Example: Configuring BPDU guard recovery time

The following example configures the BPDU guard recovery time.

```
--{ + candidate shared default }--[  ]--
# info with-context system protocols stp
    system {
        protocols {
            stp {
                bpdu-guard-recovery-time 5
            }
        }
    }
```

4.2 VXLAN v4

This chapter describes the implementation for VXLAN tunnels that use IPv4 in the underlay.

4.2.1 VXLAN configuration

VXLAN on SR Linux uses a tunnel model where VXLAN interfaces are bound to network instances, in the same way that subinterfaces are bound to network instances. Up to one VXLAN interface per network instance is supported.

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

Configuration of VXLAN on SR Linux is tied to EVPN. VXLAN configuration consists of the following steps:

1. Configure a tunnel interface and VXLAN interface.

   A tunnel interface for VXLAN is configured as `vxlan<N>`, where N can be 0 to 255.

   A VXLAN interface is configured under a tunnel interface. At a minimum, a VXLAN interface must have an index, type, and the ingress VXLAN network identifier (VNI).

   - The index can be a number in the range 0 to 4294967295.
   - The type can be bridged or routed and indicates whether the vxlan-interface can be linked to a MAC-VRF (bridged) or IP-VRF (routed).
   - The system looks for the ingress VNI in incoming VXLAN packets to classify them to this VXLAN interface and its network instance.

   Configuration of an explicit VXLAN interface egress source IP is not permitted, given that the data path supports one source tunnel IP address for all VXLAN interfaces. The source IP used in the VXLAN interfaces is the IPv4 address of sub-interface `system0.0` in the default network instance.

2. Associate the VXLAN interface to a network instance.

   A VXLAN interface can only be associated with one network instance, and a network instance can have only one VXLAN interface.

3. Associate the VXLAN interface to a BGP-EVPN instance.

   The VXLAN interface must be linked to a BGP-EVPN instance so that VXLAN destinations can be dynamically discovered and used to forward traffic.

The following configuration example shows these steps:

```
tunnel-interface vxlan1 {
  // (Step 1) Creation of the tunnel-interface and vxlan-interface
    vxlan-interface 1 {
        type bridged
        ingress {
            vni 1
        }
        egress {
            source-ip use-system-ipv4-address
        }
    }
}
network-instance blue {
    type mac-vrf
    admin-state enable
    description "network instance blue"
    interface ethernet-1/2.1 {
    }
    vxlan-interface vxlan1.1 {
  // (Step 2) Association of the vxlan-interface to the network-instance
    }
    protocols {
        bgp-evpn {
            bgp-instance 1 {
                admin-state enable
                vxlan-interface vxlan1.1
  // (Step 3) Association of the vxlan-interface to the bgp-evpn instance
                evi 1
            }
        }
        bgp-vpn {
            bgp-instance 1 {
```


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

```
                route-distinguisher {
                    route-distinguisher 1.1.1.1:1
                }
                route-target {
                    export-rt target:1234:1
                    import-rt target:1234:1
                }
            }
        }
    }
}
```

When EVPN routes are received with VXLAN encapsulation, the SR Linux creates VXLAN Termination Endpoints (VTEPs) from the EVPN route next-hops, and each VTEP is allocated an index number (per source and destination tunnel IP addresses).

When a VTEP is created in the vxlan-tunnel table and a non-zero index allocated, a tunnel-table entry is also created for the tunnel in the tunnel-table.

If the next hop is not resolved to any route in the network-instance default route-table, the index in the vxlan-tunnel table shows as `0` for the VTEP, and no tunnel-table entry would be created in the tunnel-table for that VTEP.

The following example shows the created vxlan-tunnel entries and tunnel-table entries on reception of IMET routes from three different PEs.

```
--{ + candidate shared default }--[  ]--
A:root@leaf3# info from state with-context tunnel vxlan-tunnel vtep *
    tunnel {
        vxlan-tunnel {
            vtep 100.0.0.1 {
                index 433965152188
 // index allocated per source and destination tunnel IP addresses.
 Index of 0 would mean that 10.22.22.2 is not resolved in the route-table
 and no tunnel-table entry is created.
                last-change "2025-10-07T11:34:01.000Z (27 minutes ago)"
            }
            vtep 100.0.0.2 {
                index 433965152189
                last-change "2025-10-07T11:34:01.000Z (27 minutes ago)"
            }
            vtep 100.0.0.4 {
                index 433965152190
                last-change "2025-10-07T11:34:01.000Z (27 minutes ago)"
            }
            vtep 100.0.0.111 {
                index 433965152197
                last-change "2025-10-07T11:34:06.000Z (27 minutes ago)"
            }
            vtep 100.0.0.222 {
                index 433965152199
                last-change "2025-10-07T11:34:10.000Z (26 minutes ago)"
            }
        }
    }

--{ + candidate shared default }--[  ]--
A:root@leaf3# info from state network-instance default tunnel-table ipv4
    tunnel 100.0.0.1/32 type vxlan owner vxlan_mgr id 1 {
 // tunnel table entry for VTEP 100.0.0.1, created after the vxlan-tunnel vtep 100.0.0.1
        next-hop-group 433965152204
        metric 0
        preference 0
```


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

```
        last-app-update "2025-10-07T11:34:01.719Z (27 minutes ago)"
        resource-allocation-failed false
        fib-programming {
            last-successful-operation-type modify
            last-successful-operation-timestamp "2025-10-07T11:34:01.719Z (27 minutes ago)"
            pending-operation-type none
            last-failed-operation-type none
        }
        vxlan {
            destination-address 100.0.0.1
            source-address 100.0.0.3
            time-to-live 255
        }
    }
    tunnel 100.0.0.2/32 type vxlan owner vxlan_mgr id 1 {
        next-hop-group 433965152204
        metric 0
        preference 0
        last-app-update "2025-10-07T11:34:01.719Z (27 minutes ago)"
        resource-allocation-failed false
        fib-programming {
            last-successful-operation-type modify
            last-successful-operation-timestamp "2025-10-07T11:34:01.719Z (27 minutes ago)"
            pending-operation-type none
            last-failed-operation-type none
        }
        vxlan {
            destination-address 100.0.0.2
            source-address 100.0.0.3
            time-to-live 255
        }
    }
    tunnel 100.0.0.4/32 type vxlan owner vxlan_mgr id 1 {
        next-hop-group 433965152204
        metric 0
        preference 0
        last-app-update "2025-10-07T11:34:01.719Z (27 minutes ago)"
        resource-allocation-failed false
        fib-programming {
            last-successful-operation-type modify
            last-successful-operation-timestamp "2025-10-07T11:34:01.719Z (27 minutes ago)"
            pending-operation-type none
            last-failed-operation-type none
        }
        vxlan {
            destination-address 100.0.0.4
            source-address 100.0.0.3
            time-to-live 255
        }
    }
    tunnel 100.0.0.111/32 type vxlan owner vxlan_mgr id 1 {
        next-hop-group 433965152204
        metric 0
        preference 0
        last-app-update "2025-10-07T11:34:06.745Z (27 minutes ago)"
        resource-allocation-failed false
        fib-programming {
            last-successful-operation-type modify
            last-successful-operation-timestamp "2025-10-07T11:34:06.745Z (27 minutes ago)"
            pending-operation-type none
            last-failed-operation-type none
        }
        vxlan {
            destination-address 100.0.0.111
```


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

```
            source-address 100.0.0.3
            time-to-live 255
        }
    }
    tunnel 100.0.0.222/32 type vxlan owner vxlan_mgr id 1 {
        next-hop-group 433965152204
        metric 0
        preference 0
        last-app-update "2025-10-07T11:34:10.185Z (27 minutes ago)"
        resource-allocation-failed false
        fib-programming {
            last-successful-operation-type modify
            last-successful-operation-timestamp "2025-10-07T11:34:10.186Z (27 minutes ago)"
            pending-operation-type none
            last-failed-operation-type none
        }
        vxlan {
            destination-address 100.0.0.222
            source-address 100.0.0.3
            time-to-live 255
        }
    }
    statistics {
        active-tunnels 5
        total-tunnels 5
    }
    tunnel-summary {
        tunnel-type vxlan {
            active-tunnels 5
        }
    }

--{ + candidate shared default }--[  ]--
A:root@leaf3# info from state network-instance default route-table next-hop-group 433965152204
    backup-next-hop-group 0
    backup-active false
    fib-programming {
        last-successful-operation-type add
        last-successful-operation-timestamp "2025-10-07T11:34:01.719Z (28 minutes ago)"
        pending-operation-type none
        last-failed-operation-type none
    }
    next-hop 0 {
        next-hop 433965152203
 // NH ID allocated by fib_mgr for the NHG-ID
        resolved not-applicable
        resource-allocation-failed false
    }
    next-hop 1 {
        next-hop 433965152191
        resolved not-applicable
        resource-allocation-failed false
    }

--{ + candidate shared default }--[  ]--
A:root@leaf3# info from state network-instance default route-table next-hop 433965152203
    resource-allocation-failed false
    type direct
 // resolution of the NH ID
    ip-address fe80::18f5:dff:feff:3
    subinterface ethernet-1/11.0
```


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

4.2.1.1 Source and destination VTEP addresses

In the network egress direction, the vxlan-interface/egress/source IP leaf determines the loopback interface that the system uses to source VXLAN packets (outer IP SA). The source IP used in the VXLAN interfaces is the IPv4 address of subinterface `system0.0` in the default network instance.

The egress VTEP (outer IP DA) is determined by EVPN and must be of the same family IPv4 as the configured source IP.

Only unicast VXLAN tunnels are supported (outer IP DA is always unicast), and ingress replication is used to deliver BUM frames to the remote VTEPs in the current release.

In the network ingress direction, the IP DA matches one of the local loopback IP addresses in the default network instance to move the packet to the VNI lookup stage (for loopback interfaces only, not other interfaces in the default network instance, such as IRB subinterfaces). The loopback IP address does not need to match the configured source IP address in the VXLAN interface.

The system can terminate any VXLAN packet with an outer destination IP matching a local loopback address, with no set restriction on the number of IPs.

4.2.1.2 Ingress or egress VNI

The configured ingress VNI determines the value used by the ingress lookup to find the network instance for a further MAC/IP lookup. The egress VNI is specified by EVPN. The system ignores the value of the I flag on reception. According to RFC 7348, the I flag must be set to 1. However, the system accepts VXLAN packets with I flag set to 0; the I flag is set to 1 on transmission.

4.2.1.3 VLAN tagging for VXLAN

Outer VLAN tagging is supported (one VLAN tag only), assuming that the egress subinterface in the default network instance uses VLAN tagging.

Inner VLAN tagging is transparent, and no specific handling is needed at network ingress for Layer 2 network instances. Inner VLAN tagging is not supported for VXLAN-originated traffic or VXLAN-terminated traffic in IP-VRF network instances that are BGP-EVPN IFL enabled.

4.2.1.4 Network instance and interface MTU

No specific MTU checks are done in network instances configured with VXLAN. Make the default network-instance interface MTU large enough to allow room for the VXLAN overhead. If the size of the egress VXLAN packets exceeds the IP MTU of the egress subinterface in the default network instance, the packets are still forwarded. No statistics are collected, other than those for forwarded packets.

IP MTU checks are used only for the overlay domain; that is, for interfaces doing inner packet modifications. IP MTU checks are not done for VXLAN-encapsulated packets on egress subinterfaces of the default network instance (which are in the underlay domain).

4.2.1.5 Fragmentation for VXLAN traffic

Fragmentation for VXLAN traffic is handled as follows:

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

- The Don't Fragment (DF) flag is set in the VXLAN outer IP header.
- The TTL of the VXLAN outer IP header is always 255.
- No reassembly is supported for VXLAN packets.

4.2.2 VXLAN and ECMP

Unicast traffic forwarded to VXLAN destinations can be load-balanced on network (underlay) ECMP links or overlay aliasing destinations.

Network LAGs, that is, LAG subinterfaces in the default network instance, are not supported when VXLAN is enabled on the on 7220 IXR-D*x* platforms. LAG access subinterfaces on MAC-VRFs are supported on all platforms.

VXLAN-originated packets support double spraying based on overlay ECMP (or aliasing) and underlay ECMP on the default network instance.

Load-balancing is supported for the following:

- encapsulated Layer 2 unicast frames (coming from a subinterface within the same broadcast domain)
- Layer 3 frames coming from an IRB subinterface

For BUM frames, load balancing operates as follows:

- BUM supports spraying in access LAGs based on a hash. That is, BUM flows received from a VXLAN or a Layer 2 subinterface of a MAC-VRF are sprayed across egress access LAG links.
- BUM does not support spraying in underlay VXLAN next-hops. That is, BUM flows received from VXLAN or a Layer 2 subinterface of a MAC-VRF are sent to a single underlay subinterface.
- BUM VXLAN packets are sent to a single member of the next-hop group (NHG) associated with a specific VXLAN multicast destination. The chosen member is based on a hash of the NHG-ID of the VXLAN destination and the number of links in the NHG.

4.2.3 VXLAN ACLs

On 7220 IXR-D*x* platforms, you can configure system filter ACLs to drop incoming VXLAN packets for reasons such as the following:

- the source IP is not recognized
- the destination IP is not an address to be used for termination
- the default destination UDP port is not being used

SR Linux supports logging VXLAN of packets dropped by ACL policies.

A system filter ACL is an IPv4 or IPv6 ACL that is evaluated before tunnel termination has occurred and before interface ACLs have been applied. A system filter can match and drop unauthorized VXLAN tunnel packets before they are decapsulated. When a system filter ACL is created, its rules are evaluated against all transit and terminating IPv4 or IPv6 traffic that is arriving on any subinterface of the router, regardless of where that traffic entered in terms of network instance, subinterface, and so on.

The system filter matches the outer header of tunneled packets; they do not filter the payload of VXLAN tunnels. If the system-filter does not drop the VXLAN-terminated packets, only egress IRB ACLs can match the inner packets. System filters can be applied only at ingress, not egress.

See the *SR Linux Configuration Basics Guide* for information about configuring system filter ACLs.

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

4.2.4 QoS for VXLAN tunnels

When the SR Linux receives a terminating VXLAN packet on a subinterface, it classifies the packet to one of eight forwarding classes and one of three drop probabilities (low, medium, or high). On 7220 IXR-D*x* platforms, the classification is based on the following considerations:

- The outer IP header DSCP is not considered.
- If the payload packet is non-IP, the classified FC is fc0 and the classified drop probability is lowest.
- If the payload packet is IP, and there is a classifier policy referenced by the **qos classifiers vxlan-default** command, that policy is used to determine the FC and drop probability from the header fields of the payload packet.
- If the payload packet is IP, and there is no classifier policy referenced by the **qos classifiers vxlan-default** command, the default DSCP classifier policy is used to determine the FC and drop probability from the header fields of the payload packet.

When the SR Linux adds VXLAN encapsulation to a packet and forwards it out a subinterface, the inner header IP DSCP value is not modified if the payload packet is IP, even if the egress routed subinterface has a DSCP rewrite rule policy bound to it that matches the packet FC and drop probability. The outer header IP DSCP is set to a static value or copied from the inner header IP DSCP. However, this static or copied value is modified by the DSCP rewrite rule policy that is bound to the egress routed subinterface, if the rule policy exists.

Example:

You can specify a classifier policy that applies to ingress packets received from any remote VXLAN VTEP. The policy applies to payload packets after VXLAN decapsulation has been performed.

The following example specifies a VXLAN classifier policy:

```
--{ * candidate shared default }--[  ]--
# info with-context qos
    qos {
        classifiers {
            vxlan-default p1 {
                traffic-class 1 {
                    forwarding-class fc0
                }
            }
        }
    }
```

See the *7250 IXR Quality of Service Guide* for information on configuring QoS on other platforms.

4.2.4.1 Configuring a VXLAN classifier policy

Procedure

On 7220 IXR-D*x* platforms, you can specify a classifier policy that applies to ingress packets received from any remote VXLAN VTEP. The policy applies to payload packets after VXLAN decapsulation has been performed.

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

Example

The following example specifies a VXLAN classifier policy:

```
--{ * candidate shared default }--[  ]--
# info with-context qos
    qos {
        classifiers {
            vxlan-default p1 {
                traffic-class 1 {
                    forwarding-class fc0
                }
            }
        }
    }
```

4.2.5 Asymmetric VXLAN VNIs

Note: This feature is supported on 7220 IXR-D2/D2L/D3/D3L/D4/D5, 7250 IXR Gen 2, and 7250 IXR Gen 2c+ platforms.

Asymmetric VNIs refer to the capability of a network instance to use different VNIs for ingress and egress traffic, instead of a single common VNI. Asymmetric VNIs are supported in MAC-VRF and IP-VRF network instances.

In MAC-VRF network instances, received MAC/IP, IMET, or AD per EVI routes with VNIs that do not match the local VNI are accepted.

In IP-VRF network instances, received IP prefixes or MAC/IP Advertisement routes with VNIs that do not match the local VNI are accepted.

There is only one unique VNI per VTEP per network instance. If a network instance receives multiple egress VNIs for the same VTEP, only the lowest egress VNI is programmed for the network instance toward that VTEP. The entries that are not programmed because of this VNI conflict will expose a next hop with **oper-state down-vni-conflict**. In addition, the egress VNI may not be programmed if there are insufficient hardware resources. If that is the case, the non-programmed next hop will show an `oper-state failed`. This is valid for all routes that can cause a conflict in the network instance.

```
# info detail from state platform linecard 1 forwarding-complex 0 fib-table next-hop-group
 24467420
    oper-state down
    backup-next-hop-group 0
    backup-active false
    next-hop 0 {
        next-hop 24467410
        oper-state down-vni-conflict
    }
```

4.2.6 VXLAN statistics collection

You can configure SR Linux to collect statistics for VXLAN tunnels on 7220 IXR-D2/D3/D2L/D3L platforms. By default, statistics collection for VXLAN tunnels is disabled.

System resources allocated to statistics collection are distributed among Layer 3 or IRB subinterface statistics (if configured), Layer 2 subinterface statistics, and VXLAN statistics.

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

When Layer 3 or IRB subinterfaces are configured, enabling VXLAN statistics collection reduces the amount of resources available for collecting Layer 2 subinterface statistics:

- If VXLAN statistics collection is disabled (the default), resources are available for collecting statistics for all supported Layer 2 subinterfaces.
- If VXLAN statistics collection is enabled, resources are shared between Layer 2 subinterfaces and VXLAN tunnels, reducing the Layer 2 subinterface scale.

  If the maximum scale is required for Layer 2 subinterfaces, VXLAN statistics collection must be disabled. Disabling VXLAN statistics collection deallocates the resources for VXLAN statistics collection and allocates them to Layer 2 subinterface statistics collection. The VXLAN statistics are not cleared, however; to clear them, use the appropriate **tools** command. See Clearing VXLAN statistics.
- The amount of resources available for collecting Layer 3 or IRB subinterface statistics does not change if VXLAN statistics collection is enabled or disabled.

4.2.6.1 Enabling VXLAN statistics collection

Procedure

To enable VXLAN statistics collection, use the following command. Enabling VXLAN statistics collection disables and re-enables statistics collection on the subinterfaces. During this transition, statistics are not counted for the subinterfaces.

After enabling VXLAN statistics collection, if you attempt to configure more than the maximum number of Layer 2 subinterfaces supported with VXLAN statistics, an error message is displayed. If more than the maximum number of Layer 2 subinterfaces with VXLAN statistics are already configured in the system, VXLAN statistics collection cannot be enabled.

Example

```
--{ candidate shared default }--[  ]--
# info with-context tunnel vxlan-tunnel statistics
    tunnel {
        vxlan-tunnel {
            statistics {
                admin-state enable
            }
        }
    }
```

4.2.6.2 Displaying VXLAN statistics

Procedure

To display statistics for all VXLAN tunnels or for a specified VTEP, use the **info from state** command in candidate or running mode, or the **info** command in state mode.

Example: Display statistics for all VXLAN tunnels

```
--{ running }--[  ]--
# info from state with-context vxlan-tunnel statistics
    vxlan-tunnel {
        statistics {
            in-octets 7296882
```


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

```
            in-packets 83012
            in-discarded-packets 5
            out-octets 7297496
            out-packets 83007
            last-clear 2021-01-29T21:58:40.919Z
        }
    }
```

Example: Display statistics for a specified VTEP

```
--{ running }--[  ]--
# info from state with-context vxlan-tunnel vtep 10.22.22.2
    vxlan-tunnel {
        vtep {
            address 10.22.22.2
            index 677716962894
            last-change 2021-01-29T21:52:34.151Z
            statistics {
                in-octets 7296882
                in-packets 83012
                in-discarded-packets 5
                out-octets 7297496
                out-packets 83007
                last-clear 2021-01-29T21:58:40.919Z
            }
        }
    }
```

4.2.6.3 Clearing VXLAN statistics

Procedure

You can clear the statistics for all VXLAN tunnels or for a specific VTEP.

Example: Clear statistics for all VXLAN tunnels

```
--{ running }--[  ]--
# tools vxlan-tunnel statistics clear
```

Example: Clear statistics for a specific VTEP

```
--{ running }--[  ]--
# tools vxlan-tunnel vtep 10.22.22.2 clear
```

4.3 MPLS transport for EVPN services

This chapter describes the infrastructure in SR Linux required to support EVPN for MPLS services. SR Linux support for MPLS itself and a detailed description of the MPLS label manager are described in the *SR Linux MPLS Guide*.

SR Linux features an MPLS label manager process that shares the MPLS label space among client applications that require MPLS labels; these applications include not only the applications managing the different MPLS supported protocols (LDP, SR-ISIS, and so on) but also EVPN manager and network-instance manager.

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

For an application such as LDP or a BGP-EVPN instance 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.

EVPN/IP-VPN services require label blocks allocated by label manager, as follows:

- A label block to be used by EVPN/IP-VPN is requested by network-instance manager based on the configuration of the **system mpls services network-instance dynamic-label-block <name>** statement. Out of this label block, a label per IP-VRF is used.
- On 7250 IXR Gen 2c+ platforms, an additional label block must be configured to support multicast labels in MAC-VRF network-instances. This additional label block is configured with the **system mpls services network-instance dynamic-evpn-inclusive-multicast-label-block <name>** statement. This additional label block is not required on other platforms, where the multicast labels can be drawn from the dynamic label block allocated to network instances.
- EVPN manager also requires configuration of a dynamic label block to be used for EVPN applications that are not network-instance specific; for example, EVPN multihoming ESI labels. This block is configured with the **system mpls services evpn dynamic-label-block <name>** statement.
- If the router is configured for inter-AS option B or next-hop-self route reflector functions, when acting as ASBR/ABR, a new label block is required so that service-label swap operations can be programmed. These labels are drawn from the label block configured with the **network-instance default protocols bgp bgp-label bgp-vpn dynamic-label-block <name>** statement, and are used for both EVPN and IP-VPN services.

If the system is not able to allocate a label, for example for a MAC-VRF where EVPN MPLS is enabled (either for unicast or multicast), the `bgp-instance` is kept down (there is no automatic retrying) with `oper-down-reason no-mpls-label`. In this case, ensure labels are made available, then re-add the `bgp-instance` so that the system can make another attempt to allocate a label.

MPLS label allocation for speculative lookup platforms

7250 IXR Gen 3 platforms require a more granular MPLS label range allocation because of the label lookup architecture of their chipsets. In these platforms, label manager allocates labels from the following blocks that are configured:

- A label block used by EVPN-IFL/IP-VPN Layer 3 encapsulated traffic is configured using the **system mpls services network-instance dynamic-label-blocks ip <name>** command. Out of this label block, a label per IP-VRF is used.
- A label block used by Ethernet encapsulated traffic (that does not use control word) terminated in MAC-VRF network instances is configured using the **system mpls services network-instance dynamic-label-blocks ethernet <name>** command. A label per MAC-VRF is used out of this label block.
- For Ethernet traffic that uses control word, a label block needs to be configured using the **system mpls services network-instance dynamic-label-blocks ethernet-with-control-word <name>** command. A label per MAC-VRF (configured to use control word) is used out of this block.
- Additional label blocks are required for multicast labels in MAC-VRF network instances. Depending on whether the network instance is configured to use a control word, the appropriate label block must be configured using the **system mpls services network-instance dynamic-label-blocks evpn-inclusive-multicast <name>** command or the **system mpls services network-instance dynamic-**

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

label-blocks evpn-inclusive-multicast-control-word <name> command. A label per MAC-VRF is used out of these blocks.

4.4 EVPN for Layer 2 ELAN services

This chapter describes the components of EVPN for Layer 2 ELAN services on SR Linux.

4.4.1 Basic configuration for EVPN for Layer 2 ELAN services

Basic configuration of EVPN for Layer 2 ELAN services for SR Linux consists of the following:

- **VXLAN interface**

  Contains the ingress VNI of the incoming VXLAN packets associated with the vxlan-interface. This is required only when EVPN uses VXLAN tunnels; it is not required when EVPN uses MPLS tunnels.

- **MAC-VRF network instance**

  Where the VXLAN interface (if present) and the BGP-EVPN instance are attached. If VXLAN is used, only one VXLAN interface can be attached to a MAC-VRF network instance.

- **BGP-EVPN**

  Enabled in the same MAC-VRF with a minimum configuration of the EVI and the VXLAN interface or the MPLS resolution associated with it.

  The BGP instance under BGP-EVPN has an encapsulation-type leaf, which is VXLAN by default, but can be set as MPLS for EVPN MPLS services.

  For EVPN, this configuration determines whether the BGP encapsulation extended community is advertised with value VXLAN or MPLS and the value encoded in the label fields of the advertised NLRIs is a VNI or MPLS label.

  For EVPN-MPLS services, the `bgp-evpn bgp-instance 1 mpls` container supports the following additional configuration and state:

  – `next-hop-resolution allowed-tunnel-types` is a list of the MPLS transport tunnel types that can be used to resolve the imported EVPN routes in the BGP-EVPN instance. At least one type is needed so that the EVPN routes can be resolved. If an EVPN route with next hop X is received, and multiple tunnel types (included in the allowed list) can resolve X, the best tunnel in the tunnel table is used.

  – `control-word` determines whether the control-word is used when encapsulating Ethernet frames into EVPN-MPLS packets, and whether the control-word is expected when receiving EVPN-MPLS packets in the context of the MAC-VRF.

  If the route-distinguisher or route-targets/policies are not configured, the required values are automatically derived from the configured EVI as follows:

  – The route-distinguisher is derived as `<ip-address:evi>`, where the `ip-address` is the IPv4 address of the default network instance subinterface system0.0.

  – The route-target is derived as `<asn:evi>`, where the `asn` is the autonomous system configured in the default network instance.

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

The following example shows a basic EVPN-VXLAN Layer 2 configuration consisting of a VXLAN interface, MAC-VRF network instance, and BGP-EVPN configuration:

```
--{ candidate shared default }--[ ]--
# info with-context
...
    tunnel-interface vxlan1 {
        vxlan-interface 1 {
            type bridged
            ingress {
                vni 10
            }
            egress {
                source-ip use-system-ipv4-address
            }
        }
    }

// In the network-instance:

A:dut2#  network-instance blue
--{ candidate shared default }--[  ]--
# info with-context network-instance blue
    network-instance blue {
        type mac-vrf
        admin-state enable
        description "Blue network instance"
        interface ethernet-1/2.1 {
        }
        vxlan-interface vxlan1.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.1
                    evi 10
                }
            }
            bgp-vpn {
                bgp-instance 1 {
     // rd and rt are auto-derived from evi if this context is not configured
                    export-policy pol-def-1
                    import-policy pol-def-1
                    route-distinguisher {
                        route-distinguisher 64490:200
                    }
                    route-target {
                        export-rt target:64490:200
                        import-rt target:64490:100
                    }
                }
            }
       }
  }
```

The following example shows a basic EVPN-MPLS Layer 2 configuration, and the previous configuration of the label-blocks required:

```
--{ + candidate shared default }--[  ]--
A:leaf3# info with-context network-instance MAC-VRF-2
    network-instance MAC-VRF-2 {
```


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

```
      type mac-vrf
      description "MAC-VRF-2 EVPN-MPLS"
      interface ethernet-1/1.2 {
      }
      interface ethernet-1/2.2 {
      }
      interface irb0.1 {
      }
      protocols {
        bgp-evpn {
          bgp-instance 1 {
            admin-state enable
            encapsulation-type mpls
            evi 2
            ecmp 4
            mpls {
              next-hop-resolution {
                allowed-tunnel-types [
                  ldp
                  sr-isis
                ]
              }
            }
          }
        }
        bgp-vpn {
          bgp-instance 1 {
          }
        }
      }
    }

// label blocks required for EVPN and network-instances

--{ + candidate shared default }--[ network-instance MAC-VRF-2 ]--
A:leaf3# /system mpls services
--{ + candidate shared default }--[  ]--
A:leaf3# info with-context system mpls
    system {
        mpls {
            services {
                evpn {
                    dynamic-label-block range-4-evpn
                }
                network-instance {
                    dynamic-label-block range-5-services
                }
            }
        }
    }
```

Note: On 7250 IXR platforms, an additional label block for EVPN multicast labels must be configured.

4.4.1.1 EVPN Layer 2 basic routes

EVPN Layer 2 without multihoming includes the implementation of the BGP-EVPN address family and support for the following route types:

- EVPN MAC/IP route (or type 2, RT2)

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

- EVPN Inclusive Multicast Ethernet Tag route (IMET or type 3, RT3)

The MAC/IP route is used to convey the MAC and IP information of hosts connected to subinterfaces in the MAC-VRF. The IMET route is advertised as soon as BGP-EVPN is enabled in the MAC-VRF; it has the following purpose:

- auto-discovery of the remote VXLAN VTEPs or MPLS TEPs (termination endpoints) attached to the same EVI

  or BD

- creation of a VXLAN or MPLS default flooding list in the MAC-VRF so that BUM frames are replicated

Advertisement of the MAC/IP and IMET routes is configured on a per-MAC-VRF basis. The following example shows the default setting **advertise true**, which advertises MAC/IP and IMET routes. The example is for EVPN-VXLAN services, but is also valid for EVPN-MPLS.

Note: Changing the setting of the **advertise** parameter and committing the change internally flaps the BGP instance.

```
--{ candidate shared default }--[ network-instance blue protocols bgp-evpn bgp-
instance 1 ]--
# info detail
    admin-state enable
    vxlan-interface vxlan1.1
    evi 1
    ecmp 1
    default-admin-tag 0
    routes {
        next-hop use-system-ipv4-address
        mac-ip {
            advertise true
        }
        inclusive-mcast {
            advertise true
        }
    }
```

4.4.1.2 Creation of EVPN destinations based on received EVPN routes

Upon receiving EVPN routes in the context of a MAC-VRF, the router creates EVPN destinations, referred to as EVPN-VXLAN destinations or EVPN-MPLS destinations, depending on whether the BGP-EVPN instance is configured for VXLAN or MPLS encapsulation.

EVPN-VXLAN destinations

Creation of VXLAN destinations of types unicast, unicast Ethernet segment (ES), and multicast for each VXLAN interface is driven by the reception of EVPN routes.

The created unicast, unicast ES, and multicast VXLAN destinations are visible in state. Each destination is allocated a system-wide unique destination index and is an internal next-hop group ID (NHG-ID). The destination indexes for the VXLAN destinations are shown in the following example for destination 10.22.22.4, vni 1:

```
--{ [FACTORY] + candidate shared default }--[ ]--
# info from state with-context tunnel-interface vxlan1 vxlan-interface 1 bridge-table unicast-
 destinations destination * vni *
    tunnel-interface vxlan1 {
```


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

```
        vxlan-interface 1 {
            bridge-table {
                unicast-destinations {
                    destination 10.44.44.4 vni 1 {
                        destination-index 677716962904 // destination index
                        statistics {
                        }
                        mac-table {
                            mac 00:00:00:01:01:04 {
                                type evpn-static
                                last-update "16 hours ago"
                            }
                        }
                    }
                }
            }
        }
    }
--{ [FACTORY] + candidate shared default }--[ ]--
# info from state with-context network-instance blue bridge-table mac-table mac
 00:00:00:01:01:04
    network-instance blue {
        bridge-table {
            mac-table {
                mac 00:00:00:01:01:04 {
                    destination-type vxlan
                    destination-index 677716962904 // destination index
                    type evpn-static
                    last-update "16 hours ago"
                    destination "vxlan-interface:vxlan1.1 vtep:10.44.44.4 vni:1"
                }
            }
        }
    }
```

The following is an example of dynamically created multicast destinations for a VXLAN interface:

```
--{ [FACTORY] + candidate shared default }--[ ]--
# info from state with-context tunnel-interface vxlan1 vxlan-interface 1 bridge-table
 multicast-destinations
    tunnel-interface vxlan1 {
        vxlan-interface 1 {
            bridge-table {
                multicast-destinations {
                    destination 40.1.1.2 vni 1 {
                        multicast-forwarding BUM
                        destination-index 46428593833
                    }
                    destination 40.1.1.3 vni 1 {
                        multicast-forwarding BUM
                        destination-index 46428593835
                    }
                    destination 40.1.1.4 vni 1 {
                        multicast-forwarding BUM
                        destination-index 46428593829
                    }
                }
            }
        }
    }
```


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

EVPN-MPLS destinations

For EVPN-MPLS services, the state for the destinations can be displayed under the MAC-VRF, as shown in the following example:

```
--{ + candidate shared default }--[ network-instance MAC-VRF-2 protocols bgp-evpn bgp-instance
 1 mpls ]--
# info from state with-context
control-word false
bridge-table {
  ingress-multicast-mpls-label 1001
  ingress-unicast-mpls-label 1000
  unicast-destinations {
    destination 100.0.0.4 evi-label 1000 tunnel-id 303822694264 {
      destination-index 303822694290
      statistics {
        active-entries 1
        total-entries 1
        failed-entries 0
        mac-type evpn {
          active-entries 0
          total-entries 0
          failed-entries 0
        }
        mac-type evpn-static {
          active-entries 1
          total-entries 1
          failed-entries 0
        }
      }
      mac-table {
        mac 1A:01:0B:FF:00:24 {
          type evpn-static
          last-update "2024-06-27T07:55:05.000Z (9 hours ago)"
        }
      }
    }
  }
  statistics {
    active-entries 1
    total-entries 1
    failed-entries 0
    mac-type evpn {
      active-entries 0
      total-entries 0
      failed-entries 0
    }
    mac-type evpn-static {
      active-entries 1
      total-entries 1
      failed-entries 0
    }
  }
  multicast-destinations {
    multicast-limit {
      maximum-entries 1024
      current-usage 1
    }
    destination 100.0.0.4 evi-label 5000 tunnel-id 303822694264 {
      multicast-forwarding BUM
      destination-index 303822694291
    }
  }
}
```


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

```
next-hop-resolution {
  allowed-tunnel-types [
    ldp
    sr-isis
  ]
}
```

4.4.1.3 EVPN route selection

When a MAC is received from multiple sources, the route is selected based on the priority listed in MAC selection. Learned and EVPN-learned routes have equal priority; the latest received route is selected.

When multiple EVPN-learned MAC/IP routes arrive for the same MAC but with a different key (for example, two routes for MAC M1 with different route-distinguishers), a selection is made based on the following priority:

1. EVPN MACs with higher SEQ number
2. EVPN MACs with lower IP next-hop
3. EVPN MACs with lower Ethernet Tag
4. EVPN MACs with lower RD

4.4.1.4 BGP next hop configuration for EVPN routes

You can configure the BGP next hop to be used for the EVPN routes advertised for a network instance. This next hop is by default the IPv4 address configured in interface `system 0.0` of the default network instance. However, the next-hop address can be changed to any IPv4 address.

The system does not check that the configured IP address exists in the default network instance. Any valid IP address can be used as next hop of the EVPN routes advertised for the network instance, irrespective of its existence in any subinterface of the system. However, the receiver leaf nodes create their unicast, multicast and ES destinations to this advertised next-hop, so it is important that the configured next-hop is a valid IPv4 address.

When the system or loopback interface configured for the BGP next-hop is administratively disabled, EVPN still advertises the routes, as long as a valid IP address is available for the next-hop. However, in EVPN-VXLAN instances received VXLAN traffic on that interface is dropped.

The following example configures a BGP next hop to be used for the EVPN routes advertised for a network instance.

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1
network-instance MAC-VRF-1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        bridge-table {
                            next-hop 1.1.1.1
                        }
                    }
                }
            }
        }
```


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

```
    }
```

4.4.2 MAC duplication detection for Layer 2 loop prevention in EVPN

MAC loop prevention in EVPN broadcast domains is based on the SR Linux MAC duplication feature (see MAC duplication detection and actions), but considers MACs that are learned via EVPN as well. The feature detects MAC duplication for MACs moving among bridge subinterfaces of the same MAC-VRF, as well as MACs moving between bridge subinterfaces and EVPN in the same MAC-VRF, but not for MACs moving from a VTEP to a different VTEP, via EVPN, in the same MAC-VRF.

Also, when a MAC is declared as duplicate, and the **blackhole** configuration option is added to the interface, then not only incoming frames on bridged subinterfaces are discarded if their MAC SA or DA match the blackhole MAC, but also frames encapsulated in EVPN-VXLAN/MPLS packets are discarded if their source MAC or destination MAC match the blackhole MAC in the mac-table.

When a MAC exceeds the allowed number of moves, the MAC is moved to a type duplicate irrespective of the type of move: EVPN-to-local, local-to-local, local-to-EVPN. The EVPN application receives an update that advertises the MAC with a higher sequence number, which may trigger the duplication in other nodes. The duplicate MAC can be overwritten by a higher priority type, or flushed by the **tools** command (see Deleting entries from the bridge table).

4.4.3 EVPN Layer 2 multihoming

SR Linux supports single-active multihoming and all-active multihoming, as defined in RFC 7432. The EVPN multihoming implementation uses the following SR Linux features:

- **system network instance**

  A system network instance container hosts the configuration and state of EVPN for multihoming.

- **BGP network instance**

  The ES model uses a BGP instance from where the RD/RT and export/import policies are taken to advertise and process the multihoming ES routes. Only one BGP instance is allowed, and all the ESes are configured under this BGP instance. The RD/RTs cannot be configured when the BGP instance is associated with the system network instance; however, the operational RD/RTs are still shown in state.

- **ESs**

  An ES has an admin-state (disabled by default) setting that must be toggled to change any of the parameters that affect the EVPN control plane. In particular, the ESes support the following:

  – general and per-ES boot and activation timers

  – manual 10-byte ESI configuration

  – all-active and single-active multihoming modes

  – DF Election algorithm type Default (modulo based) or type Preference

  – configuration of ES and AD per-ES routes next-hop, and ES route originating-IP per ES

  – an AD per ES route is advertised per MAC-VRF, where the route carries the network-instance RD and RT

  – association with an interface that can be of type Ethernet or LAG. When associated with a LAG, the LAG can be static or LACP-based. In case of LACP, the same **system-id**, **system-priority**, and **port-key** settings must be configured on all the nodes attached to the same ES.

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

- **aliasing load balancing**

  This hashing operation for aliasing load balancing uses the hash fields that the platform uses for ECMP or LAG load balancing.

- **reload-delay timer**

  The reload-delay timer configures an interface to be shut down for a period of time following a node reboot or an IMM reset to avoid black-holing traffic.

4.4.3.1 EVPN Layer 2 multihoming procedures

EVPN relies on three different procedures to handle multihoming: DF election, split-horizon, and aliasing. DF election is relevant to single-active and all-active multihoming; split-horizon and aliasing are relevant only to all-active multihoming.

- **Designated Forwarder (DF) Election**

  The DF is the leaf that forwards BUM traffic in the ES. Only one DF can exist per ES at a time, and it is elected based on the exchange of ES routes (type 4) and the subsequent DF Election Algorithm (DF Election Alg).

  In single-active multihoming, the non-DF leafs bring down the subinterface associated with the ES.

  In all-active multihoming, the non-DF leafs do not forward BUM traffic received from remote EVPN PEs.

- **split-horizon**

  This is the mechanism by which BUM traffic received from a peer ES PE is filtered so that it is not looped back to the CE that first transmitted the frame. This mechanism is different depending on whether the transport is VXLAN or MPLS. For VXLAN, local bias is applied, as described in RFC 8365. For EVPN-MPLS packets, the split-horizon filtering is based on ESI-label based filtering, as described in RFC 7432.

- **aliasing**

  This is the procedure by which PEs that are not attached to the ES can process non-zero ESI MAC/IP routes and AD routes and create ES destinations to which per-flow ECMP can be applied.

To support multihoming, EVPN supports two additional route types:

- **ES routes (type 4)**

  Used for ES discovery on all the leafs attached to the ES and DF Election.

  ES routes use an ES-import route target extended community, its value derived from the ESI, so that its distribution is limited to only the leafs that are attached to the ES.

  The ES route is advertised with the DF Election extended community, which indicates the intent to use a specific DF Election Alg and capabilities.

  On reception of the remote ES routes, each PE builds a DF candidate list based on the originator IP of the ES routes. Then, based on the agreed DF Election Alg, each PE elects one of the candidates as DF for each MAC-VRF where the ES is defined.

- **AD route (type 1)**

  Advertised to the leafs attached to an ES. There are two versions of AD routes:

  – **AD per-ES route**

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

Used to advertise the multihoming mode (single-active or all-active) and the ESI label, which is not advertised or processed in case of EVPN-VXLAN, but it is advertised and used in the case of EVPN-MPLS. Its withdrawal enables the mass withdrawal procedures in the remote PEs.

– **AD per-EVI route**

Used to advertise the availability of an ES in an EVI and its VNI/label. It is needed by the remote leafs for the aliasing procedures.

Both versions of AD routes can influence the DF Election. Their withdrawal from a leaf results in removing that leaf from consideration for DF Election for the associated EVI, as long as **ac-df exclude** is configured. The AC-DF capability can be set to **exclude** only if the DF election algorithm type is set to **preference**.

4.4.3.2 All-active multihoming

4.4.3.2.1 EVPN-VXLAN local bias for all-active multihoming

Local bias for all-active multihoming in EVPN-VXLAN services is based on the following behavior at the ingress and egress leafs:

- At the ingress leaf, any BUM traffic received on an all-active multihoming LAG subinterface associated with an EVPN-VXLAN MAC-VRF is flooded to all local subinterfaces, irrespective of their DF or NDF status, and VXLAN tunnels.
- At the egress leaf, any BUM traffic received on a VXLAN subinterface associated with an EVPN-VXLAN MAC-VRF is flooded to single-homed subinterfaces and multihomed subinterfaces whose ES is not shared with the owner of the source VTEP if the leaf is DF for the ES.

In SR Linux, the local bias filtering entries on the egress leaf are added or removed based on the ES routes, and they are not modified by the removal of AD-per-EVI/ES routes. This configuration may cause blackholes in the multihomed CE for BUM traffic if the local subinterfaces are administratively disabled.

4.4.3.2.2 EVPN-MPLS split-horizon filtering for all-active multihoming

For Ethernet segments attached to EVPN-MPLS MAC-VRFs, EVPN manager requests a label block from which it draws the ES labels. A single label is allocated per ES on the router, and that label is advertised in the EVPN auto-discovery-per-ES route. The peer PEs attached to the same ES as the local router push the received ESI label at the bottom of the stack when sending BUM encapsulated packets to the advertising PE. The ESI label identifies BUM packets that are sourced from a given ES, and therefore cannot be forwarded to the same ES from where they came.

4.4.3.3 Single-active multihoming

Figure 6: EVPN Layer 2 single-homing configuration shows a single-active ES attached to two leaf nodes. In this configuration, the ES in single-active mode can be configured to do the following:

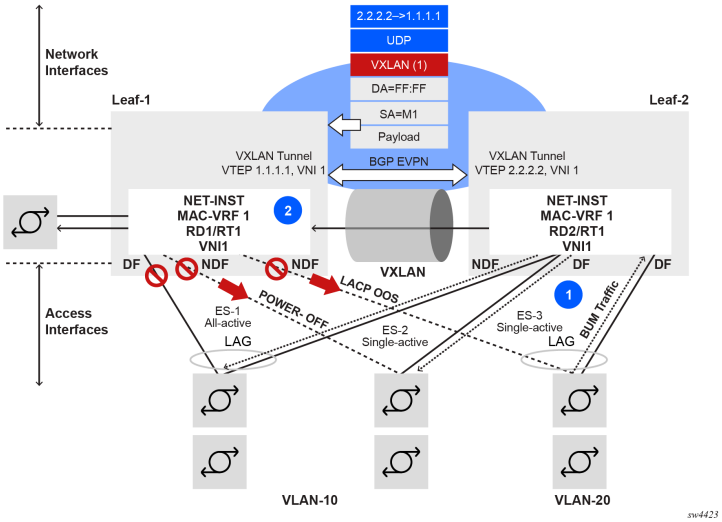
- associate with an Ethernet interface or a LAG interface, as all-active ESes
- coexist with all-active ESes on the same node, as well as in the same MAC-VRF service
- signal the non-DF state to the CE by using LACP out-of-sync signaling or power off

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

Optionally, the ES can be configured not to signal to the CE. When the LACP sync flag or power off is used to signal the non-DF state to the CE or server, all of the subinterfaces are active on the same node; that is, load balancing is not per-service, but rather per-port. This mode of operation is known as EVPN multihoming port-active mode.

- connect to a CE that uses a single LAG to connect to the ES or separate LAG/ports per leaf in the ES

Figure 6: EVPN Layer 2 single-homing configuration



All peers in the ES must be configured with the same multihoming mode; if the nodes are not configured consistently, the `oper-multi-homing-mode` in state is single-active. From a hardware resource perspective, no local-bias-table entries are populated for ESes in single-active mode.

The following features work in conjunction with single-active mode:

- Preference-based DF election/non-revertive option configures ES peers to elect a DF based on a preference value, as well as an option to prevent traffic from reverting back to a former DF node.
- Attachment Circuit-influenced DF Election (AC-DF) allows the DF election candidate list for a network instance to be modified based on the presence of the AD per-EVI and per-ES routes.
- Standby LACP-based or power-off signaling configures how the node's non-DF state is signaled to the multihomed CE.

4.4.3.3.1 Preference-based DF election/non-revertive option

Preference-based DF election is defined in *draft-ietf-bess-evpn-pref-df* and specifies a way for the ES peers to elect a DF based on a preference value where highest preference-value wins. The *draft-ietf-bess-evpn-pref-df* document also defines a non-revertive mode, so that on recovery of a former DF node, traffic does not revert to the node. This mode is desirable in most cases to avoid double impact on the traffic (failure and recovery).

The configuration requires the command **df-election/algorithm/type preference** and the corresponding **df-election/algorithm/preference-alg/preference-value**. Optionally, you can set non-revertive mode to **true**. See EVPN multihoming configuration example.

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

Nokia recommends that all of the peers in the ES be configured with the same algorithm type. However, if that is not the case, all the peers fall back to the default algorithm or operational type.

4.4.3.3.2 Attachment Circuit-influenced DF Election (AC-DF)

Attachment Circuit-influenced DF Election (AC-DF) refers to the ability to modify the DF election candidate list for a network instance based on the presence of the AD per-EVI and per-ES routes. When AC-DF is enabled with the **ac-df include** command, a node cannot become DF if it has the ES subinterface in administratively disabled state. The AC-DF capability is defined in RFC 8584, and it is by default enabled.

Nokia recommends disabling the AC-DF capability, **ac-df exclude** command, when single-active multihoming is used and standby-signaling (**lacp power-off** command) signals the non-DF state to the multihomed CE or server. In this case, the same node must be DF for all the contained subinterfaces. Administratively disabling one subinterface does not cause a DF switchover for the network instance if **ac-df exclude** is configured.

The AC-DF capability is configured with the command **df-election algorithm preference-alg capabilities ac-df**; **include** is the default. See EVPN multihoming configuration example.

4.4.3.3.3 Standby LACP-based or power-off signaling

Standby LACP-based or power-off signaling is used for cases where the AC-DF capability is excluded, and the DF election port-active mode is configured.

When single-active multihoming is used and all subinterfaces on the node for the ES must be in DF or non-DF state, the multihomed CE does not send traffic to the non-DF node. SR Linux supports two ways of signaling the non-DF state to the multihomed CE: LACP standby or power-off.

Signaling the non-DF state is configured at the interface level, using the command **interface ethernet standby-signaling**, and must also be enabled for a specific ES using the **ethernet-segment df-election interface-standby-signaling-on-non-df** command. See EVPN multihoming configuration example.

The LACP signaling method is only available on LAG interfaces with LACP enabled. When the node is in the non-DF state, it uses an LACP out-of-sync notification (the sync bit is clear in the LACP PDUs) to signal the non-DF state to the CE. The CE then brings down LACP, and the system does not jump to the collecting-distributing state and neither does the peer (because of the out_of_sync state). After the port is out of standby mode, LACP needs to be re-established, after which the forwarding ports need to be programmed.

The power-off signaling is available on Ethernet and LAG interfaces. When the node is in non-DF state, the interface goes operationally down, and the lasers on the Ethernet interfaces (all members in case of a LAG) are turned off. This process brings the CE interface down and avoids any traffic on the link. The interfaces state show `oper-state down` and `oper-down-reason standby-signaling`.

4.4.3.4 Reload-delay timer

After the system boots, the reload-delay timer keeps an interface shut down with the laser off for a configured amount of time until connectivity with the rest of network is established. When applied to an access multihomed interface, typically an ES interface), this delay can prevent black-holing traffic coming from the multihomed server or CE.

In EVPN multihoming scenarios, if one leaf in the ES peer group is rebooting, coming up after an upgrade or a failure, it is important for the ES interface not to become active until after the node is ready to forward

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

traffic to the core. If the ES interface comes up too quickly and the node has not programmed its forwarding tables yet, traffic from the server is black-holed. To prevent this from happening, you can configure a reload-delay timer on the ES interface so that the interface does not become active until after network connectivity is established.

When a reload-delay timer is configured, the interface port is shut down and the laser is turned off from the time that the system determines the interface state following a reboot or reload of the XDP process until the number of seconds specified in the reload-delay timer elapse.

The reload-delay timer is only supported on Ethernet interfaces that are not enabled with breakout mode. For a multihomed LAG interface, configure the reload-delay timer on all the interface members. The reload-delay timer can be from 1 to 86 400 seconds. There is no default value; if not configured for an interface, there is no reload-delay timer.

Only configure ES interfaces with a non-zero reload-delay timer. Do not configure a reload-delay timer on single-homed interfaces and network interfaces (used to forward EVPN traffic).

The following example sets the reload-delay timer for an interface to 20 seconds. The timer starts following a system reboot or when the IMM is reconnected, and the system determines the interface state. During the timer period, the interface is deactivated and the port laser is inactive.

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/1
    interface ethernet-1/1 {
        admin-state enable
        ethernet {
            reload-delay 20
        }
    }
```

When the reload-delay timer is running, the `port-oper-down-reason` for the port is shown as `interface-reload-timer-active`. The `reload-delay-expires` state indicates the amount of time remaining until the port becomes active. For example:

```
--{ * candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/1
    interface ethernet-1/1 {
        description eth_seg_1
        admin-state enable
        mtu 9232
        loopback-mode false
        ifindex 671742
        oper-state down
        oper-down-reason interface-reload-time-active
        last-change "51 seconds ago"
        linecard 1
        forwarding-complex 0
        vlan-tagging true
        ...
        ethernet {
            auto-negotiate false
            lacp-port-priority 32768
            port-speed 100G
            hw-mac-address 00:01:01:FF:00:15
            reload-delay 20
            reload-delay-expires "18 seconds from now"
            flow-control {
                receive false
                transmit false
            }
```


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

```
        }
    }
```

4.4.3.5 EVPN multihoming for IPv6 and non-system IPv4 next-hops

Note: EVPN multihoming for IPv6 and non-system IPv4 next-hops is supported on 7250 IXR and 7220 IXRplatforms.

EVPN routes received with an IPv6 next-hop can be resolved if a matching IPv6 tunnel exists in the IPv6 tunnel table. However, when advertising EVPN routes for a VPWS, the system IPv4 address is used by default.

The **network-instance protocols bgp-evpn bgp-instance routes bridge-table next-hop** command can be used to change the next-hop to an IPv6 or non-system IPv4 address. This command can be used for both MAC-VRF and VPWS network instances. VPWS can use this command to change the route next-hops, while the rest of the bridge table commands are not configurable for VPWS. An export policy can also replace the next-hop with any IPv6 address.

When using IPv6 next-hops in EVPN VPWS, multihoming requires the modification of the ES route **originating-ip** and AD per ES routes next-hops to be consistent with the next-hop used in the network instance. If the ES and AD per ES routes are advertised with the system IPv4 address as the next-hops, the next-hops never match the ones in the service itself and the DF Election procedure does not work properly.

In order for EVPN services to support IPv6 tunnels, the following conditions must be met:

- The EVPN routes must modify the next-hop to the system IPv6 or a specific IPv6 address by changing the **network-instance protocols bgp-evpn bgp-instance routes bridge-table next-hop** command.
- EVPN multihoming routes can use IPv6 addresses by changing the following commands in the ES routes container:
  - **originating-ip**
  - **next-hop**

IPv6 or non-system IPv4 next-hop addresses are supported for network instances that use MPLS tunnels.

4.4.3.6 EVPN multihoming configuration example

The following is an example of a single-active multihoming configuration, including standby signaling power-off, AC-DF capability, and preference-based DF algorithm.

The following configures power-off signaling and the reload delay timer for an interface:

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/21
    interface ethernet-1/21 {
        ethernet {
            standby-signaling power-off // needed to signal non-DF state to the CE
            reload-delay 100 // upon reboot, this is required to avoid attracting traffic
from the multi-homed CE until the node is ready to forward.
// The time accounts for the time it takes all network protocols to be up
and forwarding entries ready.
```


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

The following configures DF election settings for the ES, including preference-based DF election and a preference value for the DF election alg. The **ac-df** setting is set to **exclude**, which disables the AC-DF capability. The **non-revertive** option is enabled, which prevents traffic from reverting back to a former DF node when the node reconnects to the network.

```
--{ * candidate shared default }--[  ]--
# info with-context system network-instance protocols
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment eth_seg_1 {
                                admin-state enable
                                esi 00:01:00:00:00:00:00:00:00:00
                                interface ethernet-1/21
                                multi-homing-mode single-active
                                df-election {
                                    interface-standby-signaling-on-non-df { // presence
 container that enables the standby-signaling for the ES
                                    }
                                    algorithm {
                                        type preference // enables the use of preference based
 DF election
                                        preference-alg {
                                            preference-value 100 // changes the default 32767
 to a specific value
                                            capabilities {
                                                ac-df exclude // turns off the default ac-df
 capability
                                                non-revertive true // enables the non-revertive
 mode
                                            }
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

The following shows the state, and consequently the configuration, of an ES for single-active multihoming and indicates the default settings for the algorithm or operational type. Nokia recommends configuring all of the peers in the ES with the same algorithm type. However, if that is not the case, all the peers fall back to the default algorithm.

```
--{ * candidate shared default }--[ system network-instance protocols evpn ethernet-
segments bgp-instance 1 ethernet-segment eth_seg_1 ]--
# info
    admin-state enable
    oper-state up
    esi 00:01:00:00:00:00:00:00:00:00
    interface ethernet-1/21
    multi-homing-mode single-active
    oper-multi-homing-mode single-active // oper mode may be different if not all
the ES peers are configured in the same way
    df-election {
```


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

```
        interface-standby-signaling-on-non-df {
        }
        algorithm {
            type preference
            oper-type preference // if at least one peer in the ES is in type
default, all the peers will fall back to default
            preference-alg {
                preference-value 100
                capabilities {
                    ac-df exclude
                    non-revertive true
                }
            }
        }
    }
    routes {
        next-hop use-system-ipv4-address
        ethernet-segment {
            originating-ip use-system-ipv4-address
        }
    }
    association {
        network-instance blue {
            bgp-instance 1 {
                designated-forwarder-role-last-change "2 seconds ago"
                designated-forwarder-activation-start-time "2 seconds ago"
                designated-forwarder-activation-time 3
                computed-designated-forwarder-candidates {
                    designated-forwarder-candidate 40.1.1.1 {
                        add-time "2 seconds ago"
                        designated-forwarder true
                    }
                    designated-forwarder-candidate 40.1.1.2 {
                        add-time "2 minutes ago"
                        designated-forwarder false
                    }
                }
            }
        }
    }
}
```

To display information about the ES, use the **show system network-instance ethernet-segments** command. For example:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# show system network-instance ethernet-segments eth_seg_1
-------------------------------------------------------------------------------
eth_seg_1 is up, single-active
  ESI      : 00:01:00:00:00:00:00:00:00:00
  Alg      : preference
  Peers    : 40.1.1.2
  Interface: ethernet-1/21
  Network-instances:
     blue
      Candidates : 40.1.1.1 (DF), 40.1.1.2
      Interface : ethernet-1/21.1
-------------------------------------------------------------------------------
Summary
 1 Ethernet Segments Up
 0 Ethernet Segments Down
-------------------------------------------------------------------------------
```


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

The **detail** option displays more information about the ES. For example:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# show system network-instance ethernet-segments eth_seg_1 detail
===============================================================================
Ethernet Segment
===============================================================================
Name                : eth_seg_1
40.1.1.1 (DF)
Admin State         : enable              Oper State        : up
ESI                 : 00:01:00:00:00:00:00:00:00:00
Multi-homing        : single-active          Oper Multi-homing : single-active
Interface           : ethernet-1/21
ES Activation Timer : None
DF Election         : preference          Oper DF Election  : preference

Last Change         : 2021-04-06T08:49:44.017Z
===============================================================================
 MAC-VRF    Actv Timer Rem   DF
eth_seg_1   0               Yes
-------------------------------------------------------------------------------
DF Candidates
-------------------------------------------------------------------------------
Network-instance       ES Peers
blue                   40.1.1.1 (DF)
blue                   40.1.1.2
===============================================================================
```

On the DF node, the **info from state** command displays the following:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/21 | grep oper
        oper-state up
            oper-state not-present
            oper-state up

--{ [FACTORY] + candidate shared default }--[  ]--
# info from state with-context network-instance blue interface ethernet-1/21.1
    network-instance blue {
        interface ethernet-1/21.1 {
            oper-state up
            oper-mac-learning up
            index 6
            multicast-forwarding BUM
        }
    }
```

On the non-DF node, the **info from state** command displays the following:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# info from state with-context interface ethernet-1/21 | grep oper
        oper-state down
        oper-down-reason standby-signaling
            oper-state not-present
            oper-state down
            oper-down-reason port-down

--{ [FACTORY] + candidate shared default }--[  ]--
# info from state with-context network-instance blue interface ethernet-1/21.1
    network-instance blue {
        interface ethernet-1/21.1 {
            oper-state down
```


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

```
            oper-down-reason subif-down
            oper-mac-learning up
            index 7
            multicast-forwarding none
        }
    }
```

When the ES is used for EVPN-MPLS segments, the state also shows the allocated ESI label; for example:

```
--{ + candidate shared default }--[ system network-instance protocols evpn ethernet-segments
 bgp-instance 1 ]--
# info from state with-context ethernet-segment ES-ce31 esi-label
  esi-label 500
```

4.4.4 Layer 2 proxy-ARP/ND

Note: Layer 2 proxy-ARP/ND are supported on 7220 IXR-D*x*, 7220 IXR-H5, 7220 IXR-H6, 7250 IXR Gen 2c+, and 7250 IXR Gen 3platforms.

Proxy-Address Resolution Protocol (ARP) and proxy-neighbor discovery (ND) are Layer 2 functions supported on MAC-VRF network instances, specified in RFC 9161. Proxy-ARP/ND enables the learning of IP-to-MAC bindings so that leaf nodes can reply to ARP-requests or neighbor solicitations (NS) without having to flood those requests in the BD. The proxy-ARP/ND function supports ARP/ND flooding suppression, except on 7220 IXR-H5 and 7220 IXR-H6 platforms. The proxy-ARP/ND function also supports security protection against ARP/ND spoofing attacks in large BDs.

When proxy-ARP/ND is enabled for a MAC-VRF, a table is created that contains entries related to proxy-ARP/ND for the BD. Entries in the proxy-ARP/ND table can be of the following types:

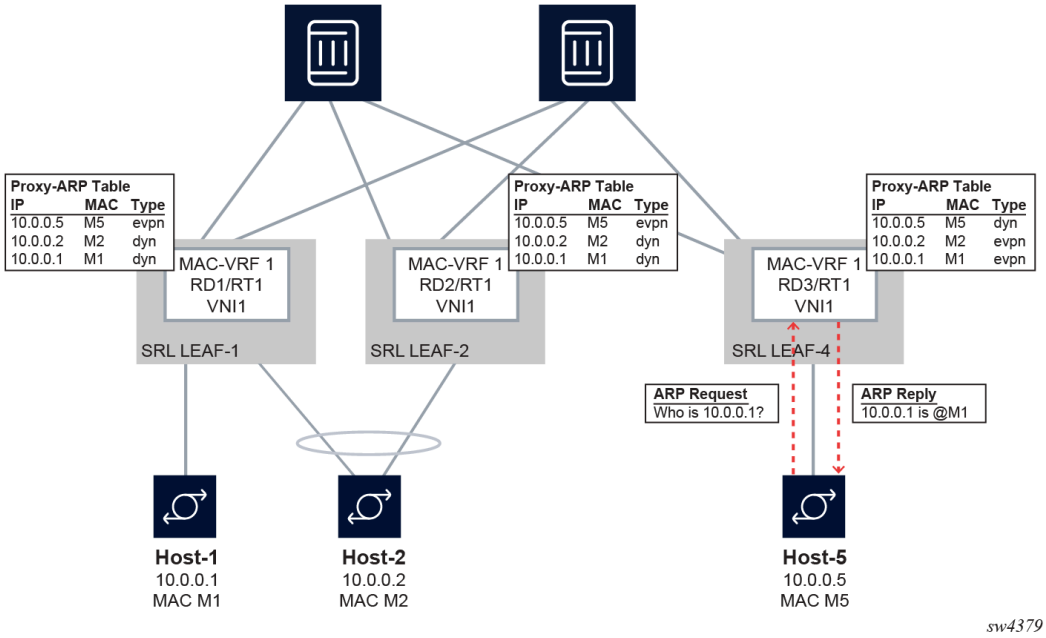
- **dynamic**
  Dynamic IP-MAC entries are learned by snooping ARP and ND messages; requires enabling proxy-ARP/ND and enabling dynamic learning. See Dynamic learning for proxy-ARP/ND.
- **static**
  Static IP-MAC entries are manually provisioned in the proxy-ARP/ND table; requires enabling proxy-ARP/ND and configuring static entries. See Static proxy-ARP entries.
- **EVPN-learned**
  EVPN-learned IP-MAC entries are learned from information received in EVPN MAC/IP (type 2 or RT2) routes from remote PE devices; requires enabling proxy-ARP/ND and configuring **bgp-evpn** on the MAC-VRF. See EVPN learning for proxy-ARP/ND.
- **duplicate**
  Duplicate entries are identified as duplicates by the IP duplication detection procedure. You can configure the criteria that determines whether an entry is a duplicate and optionally inject anti-spoofing MACs in case of duplication. See Proxy-ARP/ND duplicate IP detection.

The proxy-ARP/ND table for the MAC-VRF has a default size of 250 entries. You can modify the table size. See Proxy-ARP/ND table.

The following figure shows how proxy-ARP/ND functions in a BD with an example showing proxy-ARP.

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

Figure 7: Layer 2 proxy-ARP example



In this example, the SR Linux leaf nodes snoop and learn the local hosts and add dynamic IP-MAC entries for them in the proxy-ARP table for the MAC-VRF. The IP-MAC bindings for the local hosts are advertised in EVPN MAC/IP (type 2 or RT2) routes and installed as EVPN-learned IP-MAC entries in the proxy-ARP table on the remote SR Linux leaf nodes.

For example, SRL LEAF-1 dynamically learns the IP-MAC binding for Host-1 (10.0.0.1-M1) and adds it to the proxy-ARP table as a dynamically learned entry. The IP-MAC binding for Host-1 is advertised in an EVPN MAC/IP route. The remote SRL LEAF-4 installs the IP-MAC binding for Host-1 in its proxy-ARP table as an EVPN-learned entry. When Host-5 sends an ARP-request for 10.0.0.1, SRL LEAF-4 looks it up in the proxy-ARP table and replies with M1. Because Leaf-4 has the 10.0.0.1-M1 binding in its proxy-ARP table, it does not need to flood the request in the BD.

If the lookup is unsuccessful, the ARP-request is re-injected into the datapath and flooded in the BD. Alternatively, this flooding can be suppressed. See Configuring proxy-ARP/ND traffic flooding options.

See RFC 9161 for details about proxy-ARP/ND in EVPN deployments.

4.4.4.1 Dynamic learning for proxy-ARP/ND

When dynamic learning for proxy-ARP/ND is enabled, all frames coming into bridged subinterfaces of the MAC-VRF that have Ethertype 0x0806 (for proxy-ARP) or ICMPv6 ND (for proxy-ND) are sent to the CPM for learning. This includes ARP-request and ARP-reply (including gratuitous ARP) messages and neighbor solicitation (NS) and neighbor advertisement (NA) messages (for ND). Dynamic entries in the proxy-ARP/ND table are created from both message types. For proxy-ND, dynamic entries are created only from NA messages. Learning an entry is done irrespective of the MAC DA of the ARP or ND packet (unicast or broadcast).

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

The dynamically learned information in the table is based on the ARP/ND payload MAC SA and IP DA, and not the frame outer MAC SA, although they normally match. A valid MAC SA must be present for the entry to be learned.

In addition to learning the dynamic entry from the ARP-request, ARP-reply, or NA, the system re-injects and forwards messages as follows:

- For received ARP-request or NS messages, the system looks up the requested IP address, and if the lookup is successful, it sends an ARP-reply or NA with the MAC→IP information. If the lookup is not successful, the ARP-request/NS is re-injected and flooded based on the flood list and the configured flooding option, with source squelching. See Configuring proxy-ARP/ND traffic flooding options.

  The re-injected ARP-request or NS keeps all existing non-service-delimiting tags of the original frame. Unicast ARP-requests are replied to if there is an entry in the proxy table. If the lookup is not successful, the frame is forwarded to the MAC DA.
- For ARP-reply/NA messages, the MAC DA (unicast) is looked up in the MAC table. In case of a hit, the frame is re-injected and unicasted based on the MAC table information. If there is no hit, the frame is re-injected and flooded based on the flood list information (with source squelching). The re-injected ARP-reply/NA keeps all existing non-service-delimiting tags of the original frame.
- For ARP/ND frames that are sent to the CPM, the ARP reply/NA is always unicasted to the subinterface on which the ARP-request/NS arrived, even if the MAC itself has not yet been learned in the MAC table.

If a MAC ACL is bound to the ingress of a bridged subinterface (see the *SR Linux ACL and Traffic Steering Guide*) and proxy-ARP/ND is enabled in the network instance, the rules in the MAC ACL are applied before extraction to the CPM. Consequently, if the MAC ACL drops an ARP packet, it never reaches the CPM.

Disabling dynamic learning for proxy-ARP/ND causes the dynamically learned entries to be flushed from the proxy-ARP/ND table. You can also set an age timer (default disabled) for dynamic entries, after which they are flushed from the proxy table.

In addition, you can set a timer to refresh dynamic entries. At a configured interval (default never) the system generates ARP-requests/NS with the intent to refresh the proxy entry; if no response is received, another one is attempted.

4.4.4.1.1 Configuring dynamic learning for proxy-ARP/ND

Procedure

To configure dynamic learning for entries in the proxy-ARP table, enable proxy-ARP and dynamic learning for the MAC-VRF. The same commands exist under `proxy-nd`.

Example

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-arp
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-arp {
                admin-state enable
                dynamic-learning {
                    admin-state enable
                    age-time 600
                    send-refresh 200
                }
            }
        }
```


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

```
    }
```

This example also configures the **age-time** and **send-refresh** timers. By default, both timers are disabled.

- The **age-time** specifies in seconds the aging timer for each proxy-ARP/ND entry. When the aging expires, the entry is flushed. The age is reset when a new ARP/GARP/NA for the same IP-MAC binding is received.
- The **send-refresh** timer sends ARP-request/NS messages at the configured time so that the owner of the IP address can reply and therefore refresh its IP-MAC (proxy-ARP/ND) and MAC (FDB) entries.

4.4.4.2 Static proxy-ARP entries

You can configure static entries in the proxy-ARP/ND table. Static entries have higher priority than snooped dynamic and EVPN entries in the table.

A static proxy-ARP/ND entry requires a static or dynamic MAC entry in the MAC table to become active. The static entries are advertised in MAC/IP routes, with the MAC Mobility extended community following the information associated with the MAC entry (static bit or sequence number).

The system does not validate the MAC addresses configured in static entries.

When **proxy-arp** or **proxy-nd** is disabled, the configured static entries remain in the proxy table, unlike dynamically learned and EVPN-learned entries, which are flushed from the table.

4.4.4.2.1 Configuring static proxy-ARP entries

Procedure

To configure static proxy-ARP, add entries to the proxy-ARP table.

Example

The following example enables proxy-ARP and configures a static entry in the proxy-ARP table.

Note: The system does not validate MAC addresses specified in static entries.

The same commands exist under `proxy-nd`, although the configured IP address is an IPv6 address.

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-arp
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-arp {
                admin-state enable
                static-entries {
                    neighbor 101.1.1.1 {
                        link-layer-address 00:00:64:01:01:01
                    }
                }
            }
        }
    }
```


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

4.4.4.3 EVPN learning for proxy-ARP/ND

When proxy-ARP/ND is configured in an EVPN, the PE devices dynamically learn IP-MAC bindings for their local hosts and advertise them in EVPN MAC/IP routes to remote PE devices. The remote PE devices add these IP-MAC bindings to the proxy-ARP/ND table as EVPN-learned entries.

When a host sends an ARP-request or NS for a host on the remote side of the EVPN, if the PE device has the EVPN-learned IP-MAC binding in its proxy-ARP/ND table, it sends back an ARP-reply or NA with the remote host's MAC. If the IP-MAC binding does not exist in the PE device's proxy-ARP/ND table, the ARP-request or NS is flooded in the BD (ARP-request/NS flooding can be optionally disabled). See Figure 7: Layer 2 proxy-ARP example for an illustration of this process.

4.4.4.3.1 Configuring EVPN learning for proxy-ARP/ND

Procedure

To configure EVPN learning for proxy-ARP/ND, enable proxy-ARP/ND for the MAC-VRF and configure **bgp-evpn** to advertise the IP-MAC bindings in EVPN MAC/IP Advertisement routes and learn new IP-MAC bindings from the imported MAC/IP Advertisement routes.

Note: The **advertise-arp-nd-only-with-mac-table-entry** command is always enabled and is not configurable.

Example: Advertise local MACs and local MAC-IP pairs

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-arp
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-arp {
                admin-state enable
                }
            }
        }
    }
```

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bgp-evpn bgp-instance 1 routes bridge-table
    network-instance MAC-VRF-1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        bridge-table {
                            mac-ip {
                                advertise true
                            }
                        }
                    }
                }
            }
        }
    }
```


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

Example: Advertise local MAC-IP pairs only with a local MAC in the MAC table

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-arp
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-arp {
                admin-state enable
                }
            }
        }
    }
```

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bgp-evpn bgp-instance 1 routes bridge-table
    network-instance MAC-VRF-1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        bridge-table {
                            mac-ip {
                            }
                        }
                    }
                }
            }
        }
    }
```

Note: EVPN MAC/IP routes advertised for Layer 2 and Layer 3 differ in that Layer 3 ARP entries are always advertised in non-zero IP MAC/IP routes, while Layer 2 proxy-ARP/ND entries are advertised only when there is a local MAC in the MAC table, because the proxy-ARP/ND entry must be active.

4.4.4.4 Configuring proxy-ARP/ND traffic flooding options

Procedure

Note: The `unknown-arp-req` and `gratuitous-arp` options are not supported on 7220 IXR-H5 and 7220 IXR-H6 platforms.

If a lookup in the proxy-ARP/ND table is unsuccessful, by default the system re-injects the ARP-request into the data path and floods it in the BD.

For proxy-ARP, you can configure the following options for how ARP frames are flooded into the EVPN network. The default for both of these options is **true**.

- `unknown-arp-req`
  This option configures whether unknown broadcast ARP-requests are flooded to EVPN destinations. Unknown in this context means the lookup in the proxy-ARP/ND table was unsuccessful. Non-broadcast ARP-requests are not affected by this option.
- `gratuitous-arp`

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

This option configures whether Gratuitous ARP (GARP) requests or replies are flooded to EVPN destinations. GARPs are ARP messages where the sender's IP address matches the target's IP address. Normally the MAC DA is a broadcast address.

For proxy-ND, you can configure the following options for how NS and NA frames are flooded into the EVPN network. The default for these options is **true**.

- `unknown-neighbor-advertise-host`
  This option configures whether to flood NA replies for type `host` into the EVPN network.
- `unknown-neighbor-advertise-router`
  This option configures whether to flood NA replies for type `router` into the EVPN network.
- `unknown-neighbor-solicitation`
  This option configures whether to flood NS messages (with source squelching) into the EVPN network

Example: Disable EVPN flooding for unknown broadcast ARP-requests and GARPs

The following example disables flooding to EVPN destinations for both unknown broadcast ARP-requests and GARPs:

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-arp evpn flood
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-arp {
                evpn {
                    flood {
                        unknown-arp-req false
                        gratuitous-arp false
                    }
                }
            }
        }
    }
```

Example: Disable EVPN flooding for Neighbor Advertisements

The following example disables flooding to EVPN destinations for NA replies for type `router` and type `host`:

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-nd evpn flood
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-nd {
                evpn {
                    flood {
                        unknown-neighbor-advertise-router false
                        unknown-neighbor-advertise-host false
                    }
                }
            }
        }
    }
```


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

4.4.4.5 Proxy-ARP/ND duplicate IP detection

Proxy-ARP/ND duplicate IP detection is a security mechanism described in RFC 9161 to detect ARP/ND-spoofing attacks. In an ARP/ND-spoofing attack, an attacker sends false ARP/ND messages into a BD, with the goal of associating the attacker's IP address with a target host and directing traffic for that host to the attacker.

The proxy-ARP/ND duplicate IP detection feature monitors changes to active entries in the proxy-ARP/ND table. When an IP move occurs, for example, IP1→MAC1 is replaced by IP1→MAC2 in the table, a **monitoring-window** timer is started for a default 3 minutes. If a specified number of IP moves (default 5) is detected before the **monitoring-window** timer expires, the IP is considered to be a duplicate.

When the system detects an IP move in the proxy table, for example, IP1→MAC1 changing to IP1→MAC2, it places the IP1→MAC2 proxy table entry in pending-confirmation state for a maximum of 30 seconds. During the pending-confirmation period, the ARP/ND entry is inactive, and a confirm-message is unicast to MAC1. If no reply from MAC1 is received during the pending-confirmation period, the IP1→MAC2 entry is confirmed as legitimate and becomes active. If a reply from MAC1 is received, then MAC2 is sent a confirm-message. If MAC2 replies, an additional confirm-message is sent to MAC1. If both MAC1 and MAC2 keep replying to the confirm-messages, it triggers the duplicate IP detection procedure for IP1, because the number of IP moves exceeds the maximum allowed during the monitoring window.

When an IP is detected as a duplicate, the proxy table cannot be updated with new dynamic or EVPN-learned entries for the same IP, although you can configure a static entry for the IP. The duplicate IP is subject to this restriction until a configured **hold-down-time** expires, of which the default is 9 minutes, after which the entry for the IP is removed from the proxy table, and the monitoring process for the IP is restarted.

Anti-Spoofing MAC

You can configure an Anti-Spoofing MAC (AS-MAC). If an AS-MAC is configured, the system associates the duplicate IP with the AS-MAC in the proxy-ARP/ND table. A GARP or unsolicited-NA message with IP1→AS-MAC is sent to the local CEs, and an EVPN MAC/IP route with IP1→AS-MAC is sent to the remote PEs. This process updates the ARP/ND caches on the CEs in the BD so that CEs in the BD use the AS-MAC as MAC DA when sending traffic to IP1.

If you configure the `static-blackhole true` option, the AS-MAC is installed in the MAC table as a blackhole MAC, which discards incoming frames with a MAC source or destination matching the AS-MAC.

4.4.4.5.1 Configuring proxy-ARP/ND duplicate IP detection

Procedure

To configure proxy-ARP/ND duplicate IP detection, set the following options:

- `monitoring-window`
  This is the number of minutes the system monitors a proxy-ARP/ND table entry following an IP move (default 3 minutes).
- `num-moves`
  This is the maximum number of IP moves a proxy-ARP/ND table entry can have during the `monitoring-window` before the IP is considered duplicate (default 5 IP moves).
- `hold-down-time`

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

This is the number of minutes from the time an IP is declared duplicate to the time the IP is removed from the proxy-ARP/ND table (default 9 minutes).

- `anti-spoof-mac`
  If configured, this replaces the MAC of the duplicate IP in the proxy-ARP/ND table. The AS-MAC is advertised in EVPN to remote PEs.
- `static-blackhole`
  If this option is set to `true`, this installs the AS-MAC in the MAC table as a blackhole MAC, causing incoming frames with a MAC source or destination matching the AS-MAC to be discarded.

Example: Configure proxy-ARP duplicate IP detection

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-arp ip-duplication
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-arp {
                ip-duplication {
                    monitoring-window 5
                    num-moves 7
                    hold-down-time 10
                    anti-spoof-mac 00:CA:FE:CA:FE:08
                    static-blackhole true
                }
            }
        }
    }
```

Example: Configure proxy-ND duplicate IP detection

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-nd ip-duplication
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-nd {
                ip-duplication {
                    monitoring-window 5
                    num-moves 7
                    hold-down-time 10
                    anti-spoof-mac 00:CA:FE:CA:FE:08
                    static-blackhole true
                }
            }
        }
    }
```

4.4.4.5.2 Configuring processing for proxy-ARP/ND probe packets

Procedure

ARP Request and NS packets that have 0.0.0.0 or :: as the sender's IP address are used as probes by the IPv4/IPv6 duplicate address detection (DAD) function. Proxy-ARP/ND identifies these probe packets and can either process them locally, or forward them into the MAC-VRF (flooding them into the MAC-VRF) so that they reach all the hosts in the BD. You can configure whether the probe packets are processed locally or flooded to the MAC-VRF.

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

Example: Configure flooding for proxy-ARP probe packets

The following example configures local processing for ARP probe messages. When set to `false`, ARP probe messages are flooded to the remote nodes if unknown-ARP-requests are configured to be flooded.

By default, local processing is set to `true`, which causes ARP probe messages used by the hosts for DAD to be processed, replied if a proxy-ARP entry is hit, or reinjected into the data path.

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1
    network-instance MAC-VRF-1 {
        type mac-vrf
        bridge-table {
            proxy-arp {
                process-arp-probes false
            }
        }
    }
```

Example: Configure flooding for Neighbor Solicitation DAD messages

The following example configures local processing for NS DAD messages. When set to `false`, NS DAD messages are flooded to the remote nodes if `unknown-neighbor-solicitation` is configured, so that unknown NS messages are flooded.

By default, local processing is set to `true`, which causes NS DAD messages used by the hosts for DAD to be processed, replied if a proxy-ND entry is hit, or reinjected into the data path.

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-nd {
                process-dad-neighbor-solicitations false
            }
        }
    }
```

4.4.4.5.3 Displaying proxy-ARP/ND duplicate IP detection information

Procedure

You can use a **show** command to display information about duplicate IPs detected in the proxy-ARP table. A similar **show** command exists for proxy-ND.

Example

To display duplicate IP information from the proxy-ARP table for all MAC-VRFs:

```
--{ running }--[  ]--
# show network-instance * bridge-table proxy-arp ip-duplication duplicate-entries
-------------------------------------------------------------------------------------------
IP-duplication in network instance mac-vrf-1
-------------------------------------------------------------------------------------------
Monitoring window       : 3 minutes
Number of moves allowed : 5
Hold-down-time          : 10 seconds
```


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

```
Anti-Spoof-MAC          : 00:DE:AD:00:00:01 (Static-blackhole)
-------------------------------------------------------------------------------------------
Duplicate entries in network instance mac-vrf-1
-------------------------------------------------------------------------------------------
+--------------+------------------------------+---------------------------+------------------+
|  Neighbor    |     MAC Address              | Detect Time               | Hold down time   |
|              |                              |                           | remaining        |
+==============+==============================+===========================+==================+
| 10.10.10.1   |  00:DE:AD:00:00:01           | 2021-12-11T12:48:24.000Z  | 10               |
| 10.10.10.2   |  00:DE:AD:00:00:01           | 2021-12-11T12:48:24.000Z  | 9                |
| 10.10.10.3   |  00:DE:AD:00:00:01           | 2021-12-11T12:48:24.000Z  | 10               |
+--------------+------------------------------+---------------------------+------------------+
-------------------------------------------------------------------------------------------
IP-duplication in network instance mac-vrf-2
-------------------------------------------------------------------------------------------
...
-------------------------------------------------------------------------------------------
Total Duplicate IPs       : 4  Total 4 Active
-------------------------------------------------------------------------------------------
```

4.4.4.6 Proxy-ARP/ND table

When you enable proxy-ARP/ND for a MAC-VRF, this creates a table containing proxy-ARP/ND entries learned dynamically by snooping ARP/ND messages, configured statically, or learned from EVPN MAC/IP routes from remote PE nodes. By default, this table can contain up to 250 entries. You can configure the size of this table to be from 1 to 8192 entries.

The system generates a log event when the size of the table reaches 90% of the maximum size and when the table reaches 95% of the maximum size. When the table reaches 100% of its maximum size, entries for an IP can be replaced, that is, a different MAC can be learned and added for the IP, but no new IP entries can be added to the table, regardless of the type (dynamic, static, or EVPN-learned).

4.4.4.6.1 Configuring the proxy-ARP/ND table size

Procedure

By default, the proxy-ARP or proxy-ND table can contain up to 250 entries of all types (dynamic, static, EVPN, duplicate). You can increase or decrease the maximum size of the table. If you configure the table size to be lower than the number of entries it currently contains, the system stops and restarts the proxy-ARP/ND application, causing the non-static entries to be flushed from the table.

Example: Configure the proxy-ARP table size

The following example configures the size of the proxy-ARP table for a MAC-VRF:

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-arp table-size
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-arp {
                table-size 125
            }
        }
    }
```


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

Example: Configure the proxy-ND table size

The following example configures the size of the proxy-ND table for a MAC-VRF:

```
--{ candidate shared default }--[  ]--
# info with-context network-instance MAC-VRF-1 bridge-table proxy-arp table-size
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-nd {
                table-size 125
            }
        }
    }
```

4.4.4.6.2 Clearing entries from the proxy-ARP/ND table

Procedure

You can use **tools** commands to clear dynamic and duplicate entries from the proxy-ARP table. You can clear all dynamic and duplicate entries or you can clear specific entries.

Example: Clear dynamic entries from the proxy-ARP table

To clear dynamic entries from the proxy-ARP table:

```
--{ running }--[  ]--
# tools network-instance MAC-VRF-1 bridge-table proxy-arp dynamic delete-all
# tools network-instance MAC-VRF-1 bridge-table proxy-arp dynamic entry 10.10.10.1 delete-
ip
```

Example: Clear duplicate entries from the proxy-ARP table

To clear duplicate entries from the proxy-ARP table:

```
--{ running }--[  ]--
# tools network-instance MAC-VRF-1 bridge-table proxy-arp duplicate delete-all
# tools network-instance MAC-VRF-1 bridge-table proxy-arp duplicate entry 10.10.10.1
 delete-ip
```

Example: Clear dynamic entries from the proxy-ND table

To clear dynamic entries from the proxy-ND table:

```
--{ running }--[  ]--
# tools network-instance MAC-VRF-1 bridge-table proxy-nd dynamic delete-all
# tools network-instance MAC-VRF-1 bridge-table proxy-nd dynamic entry 10.10.10.1 delete-
ip
```

Example: Clear duplicate entries from the proxy-ND table

To clear duplicate entries from the proxy-ND table:

```
--{ running }--[  ]--
# tools network-instance MAC-VRF-1 bridge-table proxy-nd duplicate delete-all
# tools network-instance MAC-VRF-1 bridge-table proxy-nd duplicate entry 10.10.10.1
 delete-ip
```


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

4.4.4.7 EVPN ARP/ND extended community flags

RFC 9047 defines an extended community (EC) for EVPN MAC/IP advertisement routes that carry flags related to ARP or ND. This EC includes flags that can indicate to a remote PE router whether an ARP/ND entry learned via EVPN belongs to a host or a router, or if the address is an anycast address. Information from the EC flags can allow remote PE routers configured for proxy ARP/ND to reply to local ARP or ND requests with correct information.

On SR Linux, the RFC 9047 EC is not carried in EVPN MAC/IP advertisement routes by default. You can configure SR Linux to advertise the RFC 9047 EC along with the MAC/IP routes advertised for local static, dynamic, and duplicate proxy-ARP/ND entries. When you enable SR Linux to advertise the RFC 9047 EC, you can also configure how the flags are advertised in EVPN MAC/IP routes and how the flags are set in replies to NS messages for EVPN entries.

This feature is not supported along with IRB interfaces.

The feature must be configured consistently on all PEs attached to the same BD, and in particular attached to the same ES and BD.

At the BGP level, only one EVPN ARP/ND EC is expected to be received along with an EVPN MAC/IP advertisement route. If more than one EVPN ARP/ND EC is received, the router considers only the first one on the list for processing purposes on the PE.

At the EVPN level, the flags are ignored if they come in a MAC/IP route without IP.

RFC 9047 extended community flags

RFC 9047 defines the following flags for the EVPN ARP/ND extended community. The flags field is the third octet of the EC.

- **router flag (R flag) (bit 7 in the flags field)**
  If set, the R flag indicates the IPv6→MAC pair in the MAC/IP advertisement route belongs to an IPv6 router; if not set, the IPv6→MAC pair belongs to a host.
- **override flag (O flag) (bit 6 in the flags field)**
  The O flag is normally set to 1 by the egress router when advertising IPv6→MAC pairs, set to 0 when IPv6 anycast is enabled for the BD or interface, and ignored when the received MAC/IP advertisement route belongs to an IPv4→MAC pair.

  Note: Anycast IPv6 addresses are not supported in SR Linux proxy-ND tables.

  The ingress PE installs the ND entry with the received O flag and always uses this O flag value when replying to an NS message for the IPv6 address.

  The O flag is conveyed in EVPN advertisements and advertised based on the received NA messages, and is installed in the proxy-ND entries. However, other than transferring the information between ND and EVPN, the O flag does not change any of the selection of ND entries in proxy-ND.
- **immutable flag (I flag) (bit 4 in the flags field)**
  If set, the I flag indicates the IP→MAC pair in the MAC/IP advertisement route is a configured ARP/ND entry; the IP address in the EVPN MAC/IP advertisement route can only be bound together with the MAC address specified in the same route.

  The I flag applies to IPv6 and IPv4 entries and is set in MAC/IP advertisements for static proxy-ARP/ND entries.

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

4.4.4.7.1 Enabling advertisement of the RFC 9047 extended community

When you enable advertisement of the RFC 9047 extended community, the EC is advertised along with the MAC/IP routes advertised for local static, dynamic, and duplicate proxy-ARP/ND entries. Enabling this feature also allows you to control how the EC is processed and how ARP/ND entries are selected based on the I flag.

The following example enables SR Linux to advertise the RFC 9047 EC:

```
--{ candidate shared default }--[  ]--
# info with-context network-instance m1 protocols bgp-evpn bgp-instance 1 routes bridge-table
    network-instance m1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        bridge-table {
                            mac-ip {
                                advertise-arp-nd-extended-community true
                            }
                        }
                    }
                }
            }
        }
    }
```

By default, the `advertise-arp-nd-extended-community` command is set to `false`. When this command is set to `true`, the R, I, and O flags are advertised and processed as described in the following sections:

- Advertising and processing for the R flag
- Advertising and processing for the I flag
- Advertising and processing for the O flag

Advertising and processing for the R flag

After you enable SR Linux to advertise the RFC 9047 EC, you can control the advertisement of proxy-ND EVPN entries and the reply to NS messages received for those EVPN entries by configuring the `advertise-neighbor-type` command. For example:

```
--{ candidate shared default }--[  ]--
# info with-context network-instance m1 bridge-table
    network-instance m1 {
        bridge-table {
            proxy-nd {
                evpn {
                    advertise-neighbor-type router-host
                }
            }
        }
    }
```

The `advertise-neighbor-type` command has three options: `host`, `router`, and `router-host`. The effect this command has on how SR Linux processes the R flag for static, dynamic, EVPN, and duplicate proxy-ND entries is described in Table 6: Effect the advertise-neighbor-type setting has on proxy ND entries.

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

Note the following:

- System-generated unsolicited NA messages for duplicate entries carry R = 0 or R = 1, and are flooded to EVPN if allowed by the configured flooding options.
- The R flag in the RFC 9047 EC is ignored for proxy-ARP entries.

Table 6: Effect the advertise-neighbor-type setting has on proxy ND entries

For this proxy-ND entry type	The advertise-neighbor-type setting has this effect
Static proxy-ND entries	Static proxy-ND entries are added with an R = 1 flag if configured with `neighbor <ipv6-address> type router`. NS messages received for the entry are replied to with the configured R flag. The entries are advertised in EVPN if the command `advertise-neighbor-type` is configured. If the configuration for an entry is changed from `host` to `router`, or from `router` to `host`, an unsolicited NA message is triggered containing the new flag. This unsolicited NA message is or isn't flooded into EVPN based on the configured flooding options.
Dynamic proxy-ND entries	When `proxy-nd/dynamic-learning/admin-state enable` is configured, the router learns dynamic entries from NA messages; the corresponding R flag is also learned and added to the proxy-ND table. Subsequent NS messages received for the entry are replied to with the corresponding R flag. The learned dynamic entries are advertised in EVPN depending configured `advertise-neighbor-type` setting: • `router` advertises dynamic and static entries for which R = 1 • `host` advertises dynamic and static entries for which R = 0 • `router-host` advertises dynamic and static entries for which R = 0 or R = 1 The `router-host` option is available only if `advertise-arp-nd-extended-community true` is configured. The R flag of the entry is then propagated in the R bit of the extended community for the route. • the default value is `advertise-neighbor-type router`
EVPN proxy-ND entries	The setting for `advertise-neighbor-type` controls the proxy-ND dynamic/static/duplicate entries that are advertised in EVPN: • `router` only advertises in EVPN dynamic entries learned with R = 1, or static entries configured as `router` • `host` only advertises in EVPN dynamic entries learned with R = 0, or static entries configured as `host`

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

For this proxy-ND entry type	The advertise-neighbor-type setting has this effect
	• `router-host` advertises in EVPN dynamic and static entries irrespective of the router flag, and with the correct R flag in each case. For an EVPN entry, the command adds the Router flag depending on the R flag coming in the RFC 9047 EC. • irrespective of the configured option, the absence of the RFC 9047 EC in a received MAC/IP route creates a proxy-ND entry of type R = 1 (router) The setting for `advertise-neighbor-type` must be consistent on all nodes for the same service.
Duplicate proxy-ND entries	Duplicate proxy-ND entries are treated as host or router entries as far as EVPN advertisement and response to NS messages is concerned. If `advertise-neighbor-type host` is configured, the duplicate entry is treated as host. If `advertise-neighbor-type router` or `router-host` is configured, the duplicate entry is treated as router.

Advertising and processing for the I flag

When the `advertise-arp-nd-extended-community true` command is configured,SR Linux advertises the I flag as follows:

- any static proxy-ARP/ND entry is advertised with I = 1
- duplicate entries with AS-MAC are advertised with I = 1 (in addition to O = 1 and R = 0 or 1, based on configuration)
- setting of the I flag is independent of the Static bit associated with the FDB entry; the I flag is only used with proxy-ARP/ND advertisements

On reception, the I flag is processed as follows:

When SR Linux receives an EVPN MAC/IP advertisement route containing an IP→MAC and I = 1, it installs the IP→MAC entry in the proxy-ARP/ND table as an immutable binding, overriding any existing non-immutable binding for the same IP→MAC. MAC mobility does not consider the I flag.

For multiple routes to the same IP, proxy-ARP/ND selection operates according the following rules:

- local immutable ARP/ND entries (static) are preferred first
- EVPN immutable ARP/ND entries are preferred next
- regular existing ARP/ND selection applies third

If SR Linux receives multiple EVPN MAC/IP advertisement routes with the I flag set to 1 with the same IP but a different MAC address, a route is selected using these selection rules.

If SR Linux originates an EVPN MAC/IP advertisement route containing an IP→MAC and I = 1, it also originates the route with the MAC mobility EC sticky/static flag set if the MAC is static. In this case, the MAC→IP binding is immutable, and it cannot move. If an update for the same immutable and static IP→MAC is received from a different PE, one of the two routes is selected.

There are no changes for generation of confirmation messages or GARP or unsolicited-NA messages when a new entry is learned.

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

Advertising and processing for the O flag

When the `advertise-arp-nd-extended-community true` command is configured, SR Linux advertises the O flag as follows:

- For dynamic entries, the setting for the O flag is taken from what was learned and added to the proxy-ND table.
- For static and duplicate entries, O = 1.

On reception, the O flag is processed as follows:

- The O flag is stored in the proxy-ND table, and is used when replying to a received NS message.
- For link-local neighbor advertisements, generated for solicitations to the local chassis MAC, NA messages with O = 0 are not learned.
- If an EVPN route is received without the RFC 9047 EC, the entry is installed with the default value of O = 1.
- For solicited or unsolicited NA messages, the O flag is O = 1 or O = 0, depending on the flag in the proxy-ND entry that was previously learned.

4.4.4.8 Displaying proxy-ARP/ND information

Procedure

You can use **show** commands to display information about the contents of the proxy-ARP table. Similar **show** commands exist for proxy-ND.

Example: Display the entire proxy-ARP table

To display all entries in the proxy-ARP table for a MAC-VRF:

```
--{ candidate shared default }--[  ]--
# show network-instance MAC-VRF-1 bridge-table proxy-arp all
-----------------------------------------------------------------------------------------------
Proxy-ARP table of network instance MAC-VRF-1
-----------------------------------------------------------------------------------------------
+-------------+--------------------+------------+---------+--------+--------------------------+
| Neighbor    | MAC Address        |   Type     | State   | Aging  | Last Update              |
+=============+====================+============+=========+========+==========================+
| 10.10.10.1  | 00:CA:FE:CA:FE:01  |  dynamic   | active  | 300    | 2021-12-11T12:48:24.000Z |
| 10.10.10.2  | 00:CA:FE:CA:FE:02  |  evpn      | active  | N/A    | 2021-12-11T12:48:24.000Z |
| 10.10.10.3  | 00:CA:FE:CA:FE:03  |  duplicate | active  | N/A    | 2021-12-11T12:48:24.000Z |
+-------------+--------------------+------------+---------+--------+--------------------------+
Total Static Neighbors     :    0 Total    0 Active
Total Dynamic Neighbors    :    1 Total    1 Active
Total Evpn Neighbors       :    1 Total    1 Active
Total Duplicate Neighbors  :    1 Total    1 Active
Total Neighbors            :    3 Total    3 Active
-----------------------------------------------------------------------------------------------
```

The output of this command indicates the total number of entries of each type, as well as the number that are active (replies are sent for received ARP-requests). For an entry to be considered active in the proxy-ARP table, it must have a corresponding entry in the MAC table of the type listed in the following table:

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

Table 7: MAC table entry types required for proxy-ARP table entry types to be active

Proxy-ARP table entry type	MAC table entry type
Dynamic	Learned
Dynamic	Static
Dynamic	EVPN
Static	Learned
Static	Static
EVPN	EVPN (irrespective of the ESI)
EVPN	Static or dynamic matching the EVPN ESI
Duplicate	–

If a proxy-ARP table entry has a corresponding entry in the MAC table of a type not listed in this table, then the proxy-ARP table entry is considered inactive, so no replies are sent for received ARP-requests. If the MAC-VRF is not active, its proxy-ARP table entries are not active as well.

Example: Display a specific entry in the proxy-ARP table

```
--{ candidate shared default }--[  ]--
# show network-instance MAC-VRF-1 bridge-table proxy-arp neighbor 10.10.10.1
-------------------------------------------------------------------------------
Proxy-ARP table of network instance MAC-VRF-1
-------------------------------------------------------------------------------
Neighbor                : 10.10.10.1
MAC Address             : 00:CA:FE:CA:FE:01
Type                    : dynamic
Programming Status      : Success
Aging                   : 300
Last Update             : 2021-12-11T12:48:24.000Z
Duplicate Detect time   : N/A
Hold down time remaining: N/A
-------------------------------------------------------------------------------
```

Example: Display a summary of the proxy-ARP table

```
--{ candidate shared default }--[  ]--
# show network-instance * bridge-table proxy-arp summary
-----------------------------------------------------
Network Instance Proxy-ARP Table Summary
-----------------------------------------------------
Network Instance: MAC-VRF-1
Total Static Neighbors     :    0 Total    0 Active
Total Dynamic Neighbors    :    1 Total    1 Active
Total Evpn Neighbors       :    1 Total    1 Active
Total Duplicate Neighbors  :    1 Total    1 Active
Total Neighbors            :    3 Total    3 Active
Maximum Entries  : 250
Warning Threshold: 95% (237)
Clear Warning    : 90% (225)
-----------------------------------------------------
```


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

```
Network Instance: MAC-VRF-2
Total Static Neighbors     :    1 Total    1 Active
Total Dynamic Neighbors    :    1 Total    1 Active
Total Evpn Neighbors       :    0 Total    0 Active
Total Duplicate Neighbors  :    0 Total    0 Active
Total Neighbors            :    2 Total    2 Active
Maximum Entries  : 250
Warning Threshold: 95% (237)
Clear Warning    : 90% (225)
-------------------------------------------------------
-------------------------------------------------------
Total Static Neighbors     :    1 Total    1 Active
Total Dynamic Neighbors    :    2 Total    2 Active
Total Evpn Neighbors       :    1 Total    1 Active
Total Duplicate Neighbors  :    1 Total    1 Active
Total Neighbors            :    5 Total    5 Active
-------------------------------------------------------
```

4.4.4.9 Displaying proxy-ARP/ND statistics

Procedure

To display proxy-ARP statistics, use the **info from state** command in candidate or running mode or the **info** command in state mode. You can display system-wide statistics or statistics for a specific MAC-VRF network instance.

Example: Display proxy-ARP statistics

The following example displays proxy-ARP statistics for a MAC-VRF network instance:

```
--{ candidate shared default }--[  ]--
# info from state with-context network-instance MAC-VRF-1 bridge-table proxy-arp
 statistics
    network-instance MAC-VRF-1 {
        bridge-table {
            proxy-arp {
                statistics {
                    total-entries 0
                    active-entries 0
                    in-active-entries 0
                    pending-entries 0
                    neighbor-origin {
                        origin
                        total-entries 0
                        active-entries 0
                        in-active-entries 0
                        pending-entries 0
                    }
                }
            }
        }
```

Example: Display proxy-ND statistics

The following example displays proxy-ND statistics for a MAC-VRF network instance:

```
--{ candidate shared default }--[  ]--
# info from state with-context network-instance MAC-VRF-1 bridge-table proxy-nd statistics
    network-instance MAC-VRF-1 {
        bridge-table {
```


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

```
            proxy-nd {
                statistics {
                    total-entries 0
                    active-entries 0
                    in-active-entries 0
                    pending-entries 0
                    neighbor-origin {
                        origin
                        total-entries 0
                        active-entries 0
                        in-active-entries 0
                        pending-entries 0
                    }
                }
            }
        }
```

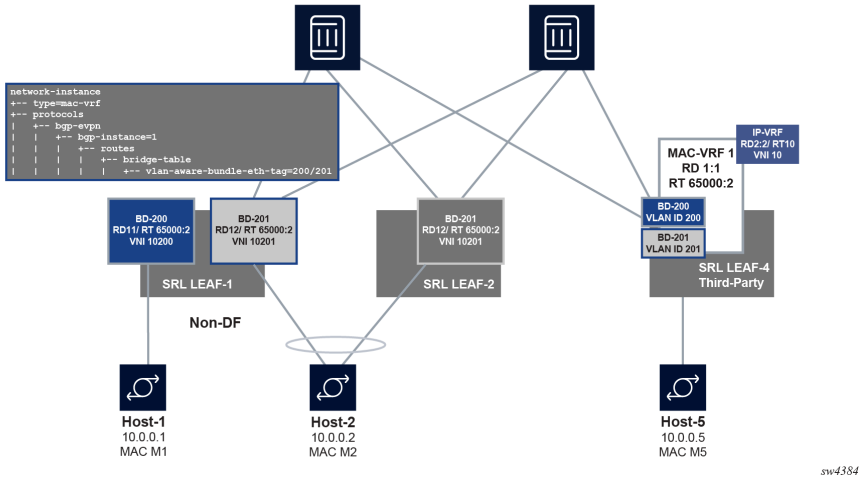
4.4.5 EVPN VLAN-aware bundle services

This chapter describes the components of EVPN VLAN-aware bundle services.

RFC 7432 defines VLAN-aware bundle services as those EVPN instances that consist of multiple broadcast domains. On SR Linux, a broadcast domain instance on a leaf node is identified by a MAC-VRF, and a MAC-VRF can contain only one broadcast domain. However, SR Linux can be attached to VLAN-aware bundle broadcast domains along with other third-party routers.

The following figure shows a configuration that features interoperability between SR Linux systems and a third-party device configured with VLAN-aware bundle services.

Figure 8: EVPN interoperability with VLAN-aware bundle services



In this configuration, Leaf-1 and Leaf-2 are SR Linux systems, and Leaf-4 is a third-party device that supports VLAN-aware bundle services. On Leaf-4, MAC-VRF1 is configured with two broadcast domains (BDs), BD-200 and BD-201. BD-200 and BD-201 are configured with the corresponding VLAN ID value encoded as Ethernet Tag ID in the EVPN routes and the VNI.

To allow Leaf-1 and Leaf-2 to interoperate with Leaf-4, the SR Linux devices are configured to advertise a non-zero Ethernet Tag ID in the EVPN routes, and process the Ethernet Tag ID in EVPN routes received

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

for the MAC-VRF. When this Ethernet Tag ID is configured, all MAC-VRFs of the same bundle have the same import/export route target (RT), but different route distinguisher (RD), Ethernet Tag ID, and VNI/label.

The Ethernet Tag ID can be set to a value in the range 0 to 16777215 (24-bit value). When the Ethernet Tag ID is set to a non-zero value, MAC/IP, IMET, and AD per-EVI routes for the MAC-VRF are advertised encoding the Ethernet Tag ID, configured with the **vlan-aware-bundle-eth-tag** parameter, into the `ethernet-tag-id` field of the routes. To interoperate in a VLAN-aware bundle broadcast domain where there are multiple vendors and multihomed CEs, all the vendors must advertise and process the AD per-EVI routes as per RFC 8584, where an AD per-EVI route is advertised per broadcast domain for the ES. SR Linux is fully compliant with RFC 8584.

For received routes, BGP processes the routes as usual, and imports them based on the import route-target. The `ethernet-tag-id` is part of the route-key, so BGP may keep multiple routes with same RD/RT/prefix but different `ethernet-tag-id`. For the routes imported in a specific MAC-VRF, only those routes whose `ethernet-tag-id` matches the locally configured Ethernet Tag ID are processed.

The feature is supported for the following:

- MAC-VRF network instances
- MAC-VRF network instances with IRB interfaces
- MAC-VRF with multihoming, for Layer 2 and Layer 3

4.4.5.1 Configuring the Ethernet Tag ID for VLAN-aware bundle services

Procedure

To configure VLAN-aware bundle services for SR Linux, you specify a value for the **vlan-aware-bundle-eth-tag** parameter.

When the **vlan-aware-bundle-eth-tag** is set to a non-zero value, routes are accepted only when the incoming `ethernet-tag-id` matches the configured value, for all route types imported in the MAC-VRF.

When the **vlan-aware-bundle-eth-tag** is set to zero (the default value):

- all received routes with `ethernet-tag-id = 0` are accepted (irrespective of type)
- for received routes with a non-zero `ethernet-tag-id` value, IMET and MAC/IP routes for the VXLAN instance are always accepted
- AD per-EVI routes with a non-zero ethernet-tag-id are rejected

Example

The following example configures a value for the **vlan-aware-bundle-eth-tag** parameter. All the EVPN routes advertised for the MAC-VRF contain this value in the `ethernet-tag-id` field.

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance mac_vrf_1 protocols bgp-evpn bgp-instance 1 routes
    network-instance mac_vrf_1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        bridge-table {
                            vlan-aware-bundle-eth-tag 1
                        }
                    }
                }
```


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

```
            }
        }
    }
```

4.4.5.2 Displaying the Ethernet Tag ID for VLAN-aware bundle services

Procedure

Use the **show network-instance** command to display the value configured for the **vlan-aware-bundle-eth-tag** parameter.

Example

```
--{ * candidate shared default }--[  ]--
# show network-instance mac_vrf_1 protocols bgp-evpn bgp-instance 1
===============================================================================================
Net Instance   : mac_vrf_1
    bgp Instance 1 is enabled and up
-----------------------------------------------------------------------------------------------
        VXLAN-Interface   : vxlan1.1
        evi               : 1
        ecmp              : 2
        default-admin-tag : 0
        oper-down-reason  : N/A
        EVPN Routes
            Next hop                      : 10.20.1.3/32 (network-instance "default" system0.0
                                            IPv4 address)
            VLAN Aware Bundle Ethernet tag : 1
            MAC/IP Routes                 : enabled
            IMET Routes                   : enabled, originating-ip 10.20.1.3/32
===============================================================================================
```

4.4.6 EVPN Proxy MAC/IP Advertisement routes in MAC-VRFs with proxy-ARP/ND

Note: This feature is supported on 7220 IXR and 7250 IXR (excluding mixed chassis) platforms.

Node failures in EVPN multi-homing (MH) environments cause significant control plane churn. In Layer 2 deployments, the issue appears as unnecessary flooding.

EVPN MH proxy MAC/IP advertisement routes on MAC-VRF reduces flooding following node failures in EVPN MH scenarios.

To configure this behavior, use the **evpn-local-es** command in the **proxy-arp** and **proxy-nd** contexts, as shown in the following example:

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance BD10
    network-instance BD10 {
        bridge-table {
            proxy-arp {
                evpn-local-es {
                    advertise true
                    retention-timer 60
                }
                evpn {
                    internal-tags {
```


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

```
                        set-tag-set [
                            1
                        ]
                    }
                }
            }
            proxy-nd {
                evpn-local-es {
                    advertise true
                    retention-timer 60
                }
                evpn {
                    internal-tags {
                        set-tag-set [
                            1
                        ]
                    }
                }
            }
        }
    }
```

EVPN local ES entries in MAC-VRF proxy-ARP/ND are advertised when `advertise true` is configured.

When **local-es advertise true** is configured, the received EVPN MAC/IP advertisement routes matching a local ESI (with an operationally up subinterface and P=0) create EVPN ARP/ND entries and are readvertised with the proxy bit set (P=1). Received routes with P=1 are not installed on nodes attached to the same operationally up ES and they are processed normally on remote nodes.

The retention timer has a default value of 0 and affects the retention of EVPN entries whose ESI matches a local ESI. When modified, 30 seconds is the minimum value.

In proxy-ARP/ND cases, the PE originating the route sets the ARP/ND Router, Override and Immutable (ROI) flags. However, for reoriginated entries, the received ROI flags in the original EVPN route are propagated in the EVPN route advertised for EVPN local ES entries. The ROI flags are supported and are propagated by the PE reoriginating the route, irrespective of the local configuration.

No separate ARP/ND entry type is created for these reoriginated entries.. The EVPN local ES entries are shown as EVPN in the show and state commands.

The advertisement of the ES local entries is supported irrespective of the MAC-VRF encapsulation.

4.4.7 Flow-aware transport label and Layer 2 attribute signaling for EVPN in MAC-VRF

Note: This feature is supported on 7250 IXR (excluding mixed chassis), 7220 IXR-D2/D2L/D3/D3L/D4/D5, and 7220 IXR-H5/H6 platforms.

Flow-aware transport (FAT) labels (also known as hash labels or flow labels) add entropy to divisible flows and create ECMP load balancing in a network. A FAT label can be used by EVPN services as per *draft-ietf-bess-rfc7432bis*.

SR Linux supports FAT labels in MAC-VRF network instances by advertising and processing the Layer 2 Attributes Extended Community on IMET routes. Signaling and processing of MTU and CW are included implicitly.

When advertisement is on and a received IMET route includes the community, the system compares the MTU, C (control word flag), and F (FAT label flag) to the local configuration.

Conflicts follow deterministic forwarding:

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

- drop the inclusive multicast destination
- avoid the matching unicast EVPN next hop
- blackhole affected remote MACs (with ES/anycast rules as specified)
- raise warning events (`evpnNetworkInstanceMtuMismatch`, `evpnNetworkInstance CwMismatch`, and `evpnNetworkInstanceFlowLabelMismatch`) and matching cleared notices when resolved
-

If IMET lacks the community or no IMET exists for a next hop, the peer is treated as not supporting Layer 2 attribute signaling; the next hop remains usable, forwarding continues, and no fault event is raised.

When **flow-label true** is configured and Layer 2 attribute advertisement is enabled, the node sets F=1 on IMET, pushes a flow label on known-unicast EVPN-MPLS toward viable peers (including peers that omit the community), and does not use flow labels on multicast or BUM paths.

Example: FAT label configuration for EVPN in MAC-VRF

The following example shows a FAT label configuration for EVPN in MAC-VRF.

- **advertise-l2-attributes-extended-community**

  When enabled, the router advertises and processes the Layer 2 Attributes Extended Community on IMET routes for the MAC-VRF BGP-EVPN instance. The default value is false.

- **ignore-mtu-mismatch**

  When enabled, a received MTU in the community does not cause an MTU-only mismatch handling. The default value is false.

- **flow-label**

  When enabled, the router signals the use of the FAT label in the F flag of the Layer 2 Attributes Extended Community sent along with the IMET route for MAC-VRF. This command cannot be enabled unless **advertise-l2-attributes-extended-community true** is configured. The default value is false.

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance mac-vrf
    network-instance mac-vrf {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    ignore-mtu-mismatch false
                    mpls {
                        flow-label true
                    }
                    routes {
                        bridge-table {
                            inclusive-mcast {
                                advertise-l2-attributes-extended-community true
                            }
                        }
                    }
                }
            }
        }
    }
```


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

4.5 EVPN for Layer 2 ELINE services

Note: These services are currently supported on the 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

SR Linux supports EVPN-VPWS services under VPWS network instances where the network instance can only use a single subinterface and BGP-EVPN.

Two connection point objects and at least one bridged subinterface must be configured. Each subinterface configured must be associated with a different connection point.

When one subinterface and EVPN network are configured, they must be associated with different connection points. The **local-attachment-circuit** and **remote-attachment-circuit** parameters must be associated with the same connection point.

The **vpws-attachment-circuits** container hosts the specific EVPN-VPWS configuration, including the configuration and state for local and remote Ethernet tags. The **vpws-attachment-circuit**s can only be configured using the **encapulation-type mpls**command.

EVPN multihoming and VPWS network instances (single and all-active) are supported on SR Linux. The configuration of the Ethernet Segments (ESs), supported features, and association with interfaces follows what is described in EVPN Layer 2 multihoming.

The main differences between multihoming in MAC-VRF and VPWS network instances are:

- VPWS services a backup PE in single-active mode, which is the next best PE after the elected DF. Remote nodes forward VPWS traffic to the primary PE (which signals flag P=1), and immediately switch over to the backup PE (which signals flag B=1) upon a failure on the primary PE.
- For MAC-VRF services, the primary/backup concept is not implemented. Upon a failure on the primary PE, the backup PE does not wait for the activation timer to activate the interface.
- VPWS services do not use the ESI label, because traffic from a PE can never be forwarded back to the ES peer PE.
- All-active multihoming for VPWS services does not run DF election, however the router still shows only four candidates in the ES state.

Entropy labels are supported on EVPN-VPWS network instances, but configuration per network instance is not supported. The entropy label is automatically added if supported by the LSP. If the EVPN destination in the VPWS service uses multiple transport LSPs, and all support entropy labels, only the inner most level uses the entropy label.

The use of **control-word** is supported.

To resolve EVPN-MPLS routes to tunnels in the tunnel-table, the network-instance must be configured with:

- **encapsulation-type mpls**
- **mpls next-hop-resolution allowed-tunnel-types[X]**, where X is at least one of the tunnel types supported by SR Linux

When EVPN routes are resolved to tunnels that support (underlay) ECMP, EVPN packets are sprayed over multiple paths for the tunnel.

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

4.5.1 Configuring EVPN-VPWS services

The following example configures a single-homing EVPN-VPWS service:

```
--{ * candidate shared default }--[  ]--
A:srl2# info with-context network-instance VPWS-4
    network-instance VPWS-4 {
        type vpws
        description "VPWS-4 EVPN-MPLS single-homing"
        interface ethernet-1/1.4 {
            connection-point A
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    encapsulation-type mpls
                    evi 4
                    ecmp 4
                    vpws-attachment-circuits {
                        local {
                            local-attachment-circuit ac-4 {
                                ethernet-tag 4
                                connection-point B
                            }
                        }
                        remote {
                            remote-attachment-circuit ac-4 {
                                ethernet-tag 4
                                connection-point B
                            }
                        }
                    }
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [
                                ldp
                            ]
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
        connection-point A {
        }
        connection-point B {
        }
    }
```

Similar to MAC-VRF network instances, VPWS network instances make use of an **oper-vpws-mtu** value derived from the **sub-interface l2-mtu** of the subinterface with lower Layer 2 MTU attached to the network instance, without the service delimiting tag size. If there is no subinterface in the VPWS network instance, then the **oper-vpws-mtu** is computed as **system mtu default-l2-mtu**.

This **oper-vpws-mtu** value is advertised in the Layer 2 information extended community along with the AD per-EVI route for the service. On reception, the local **oper-vpws-mtu** value must match the received Layer

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

2 MTU value. If the values do not match, the EVPN destination is not created. A received value of zero is ignored and the destination is created, assuming the rest of the attributes are valid.

4.5.2 Operational state in VPWS network instances

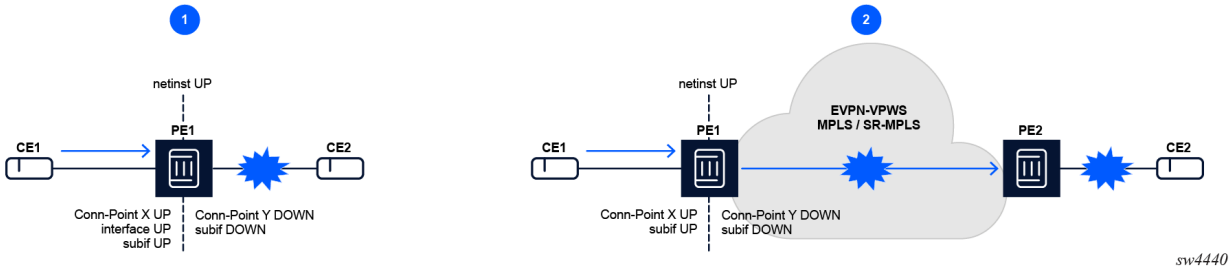
In a VPWS network instance, the following objects have their own operational state:

- **network-instance**
- **bgp-evpn bgp-instance**
- **connection-point**
- **subinterface**
- **network-instance interface**

The network instance is operationally up as long as **admin-state enable** is configured. The network instance remains up regardless of the state of the **connection-points**, **bgp-evpn bgp-instance** or the **subinterfaces**.

The **connection-point** is operationally up as long as at least one object in the **connection-point** is operationally up. This object can be a subinterface or an EVPN destination. An example of failures and how they affect the operational state of different VPWS objects is depicted in the following figure.

Figure 9: VPWS operational state



In case 1, PE1 is attached to a VPWS service with two subinterfaces. If the interface and/or subinterface CE2 goes down:

- Connection point Y goes operationally down.
- Connection point X and subinterface CE1 remain operationally up and the network instance remains operationally up.

In case 2, PE1 is attached to an EVPN-VPWS service with a subinterface and an EVPN destination. If the interface and/or subinterface CE2 goes down:

- EVPN destination on PE is removed and connection point Y goes operationally down.
- **bgp-evpn bgp-instance** remains operationally up.
- Connection point X and subinterface CE1 remain operationally up and the network instance remains operationally up. This changes when **link-loss-forwarding true** is enabled. For more information, see Interface/LAG link loss forwarding.

In a third example (not shown in the figure), the two connection points can go operationally down but the network instance remains operatioanlly up.

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

4.5.3 Layer 2 attribute mismatch

A Layer 2 attribute mismatch occurs when an AD per EVI route is imported into a VPWS service and the signaled layer 2 attributes (Layer 2 MTU, control word, or flow label) do not match the local settings.

In general, a Layer 2 attribute mismatch causes the `evpn_mgr` to discard the AD per EVI route and remove the EVPN destination from the PE that advertised the route.

The received Layer 2 MTU must match the **oper-vpws-mtu** so that the EVPN destination can be created. If the received MTU value is zero, the value is ignored and the destination is created if the rest of the attributes are valid.

A mismatch on the C flag (the flag that indicates the capability to use **control-word**) removes the EVPN destination. Even if the received C flag matches the local setting, if packets are received without the **control-word** and the **control-word** is configured, the data path discards the packets. Packets are also dropped if the **control-word** is received but **control-word false** is configured.

A mismatch on the F flag (flow label) also removes the EVPN destination.

4.5.4 Flow-aware transport label for EVPN-VPWS

Flow-aware transport (FAT) labels (also known as hash labels or flow labels) add entropy to divisible flows and create ECMP load balancing in a network. A FAT label can be used by EVPN services as per *draft-ietf-bess-rfc7432bis*.

SR Linux supports FAT labels for EVPN-VPWS services. When configured, FAT label support is always signaled as a new flag in the Layer 2 attributes extended community (F flag).

When **flow-label** is configured to be true, the EVPN AD per EVI route for the VPWS service is signaled with the F flag set and the local node expects the peer route to be received with F=1. If there is a mismatch in the F flag value between peers, the EVPN destination is removed.

Example: FAT label configuration for EVPN-VPWS

The following example shows a FAT label configuration for EVPN-VPWS.

```
--{ * candidate shared default }--[ network-instance vpws-1 protocols bgp-evpn ]--
A:srl2# info with-context
    network-instance mgmt {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    mpls {
                        flow-label true
                    }
                }
            }
        }
    }
```

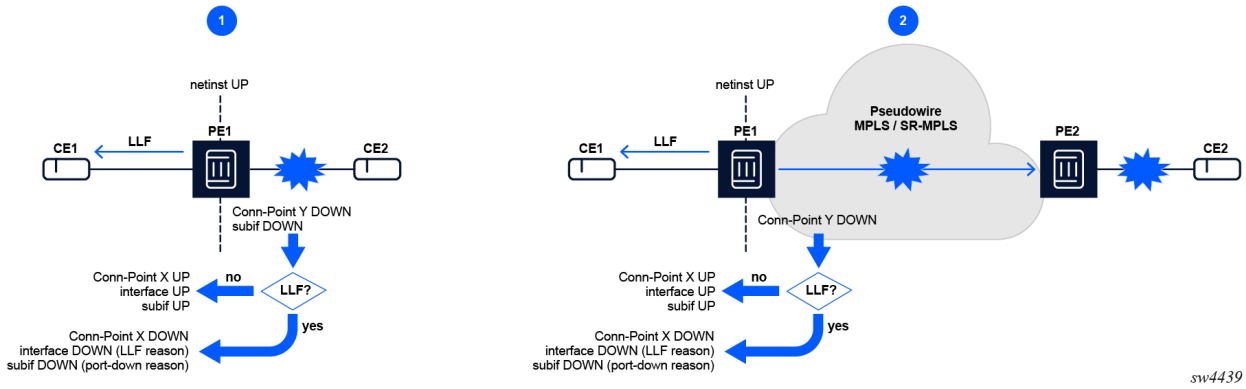

© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

4.5.5 Interface/LAG link loss forwarding

Interface/LAG link loss forwarding (LLF) covers the propagation of a fault on an EVPN-VPWS service. The removal of all the AD per EVI routes from a remote PE triggers a fault propagation event on the local PE to the local CE. That fault propagation can use power-off or LACP out-of-synch signaling.

As illustrated in the following figure, LLF works in VPWS network instances irrespective of the connection points configured with two subinterfaces, one subinterface, and BGP-EVPN.

Figure 10: LLF in VPWS network instances



In the cases 1 and 2 in the preceding figure, upon a failure that causes connection point Y to go operationally down, the system checks the configuration of the interface on connection point X. If **interface ethernet link-loss-forwarding true** is configured in the interface of connection point X, the **interface ethernet standby-signaling** action is triggered. Otherwise, connection point X and its subinterface remain operationally up.

In case 2, where BGP-EVPN is configured, when the subinterface connected to CE1 is operationally down because of active LLF, the EVPN AD per EVI route is still advertised by PE1.

LLF does not introduce any **oper-down-reason** at connection point or subinterface levels. LLF introduces **port-oper-per-down-reason link-loss-forwarding** at the interface level.

Referring to the preceding figure, suppose connection point Y goes operationally down and LLF is configured to be true. In this case the:

- connection point X goes operationally down with the **oper-down-reason** being **associations-oper-down**.
- subinterface to CE1 goes operationally down with the **oper-down-reason** being **port-down**.
- interface goes operationally down with the **oper-down-reason** being **link-loss-forwarding**.

LLF is compatible with ESs in all-active or single-active mode.

Configuring link loss forwarding

LLF is enabled by the **interface ethernet link-loss-forwarding true** command on the interface connected to the CE over which the fault signaling is propagated. In addition, the command **standby-signaling**

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

power-off lacp must be configured on the same interface so that `ch_mgr` can trigger the relevant standby signaling to the CE.

```
--{ * candidate shared default }--[  ]--
A:srl# info with-context interface ethernet-1/1
    interface ethernet-1/1 {
        ethernet {
            standby-signaling power-off
            link-loss-forwarding true
        }
    }
```

4.6 EVPN for Layer 3

Note:
The components of EVPN Layer 3 are supported on 7220 IXR-D*x*, 7220 IXR-H5, 7220 IXR-H6, 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

This chapter describes the components of EVPN Layer 3 on SR Linux.

4.6.1 EVPN Layer 3 basic configuration

The basic EVPN Layer 3 configuration model builds on the model for EVPN routes described in EVPN for Layer 2 ELAN services, extending it with an additional route type to support inter-subnet forwarding: EVPN IP prefix route (or type 5, RT5).

The EVPN IP prefix route conveys IP prefixes of any length and family (IPv4 or IPv6) that need to be installed in the ip-vrfs of remote leaf nodes. The EVPN Layer 3 configuration model has two modes of operation:

- **asymmetric IRB**

  This is a basic mode of operation EVPN Layer 3 using IRB interfaces. The term asymmetric refers to how there are more lookups performed at the ingress leaf than at the egress leaf.

  While the asymmetric model allows inter-subnet-forwarding in EVPN-VXLAN networks in a very simple way, it requires the instantiation of all the MAC-VRFs of all the tenant subnets on all the leafs attached to the tenant. Because all the MAC-VRFs of the tenant are instantiated, FDB and ARP entries are consumed for all the hosts in all the leafs of the tenant.

  These scale implications may make the symmetric model a better choice for data center deployment.

- **symmetric IRB**

  The term symmetric refers to how MAC and IP lookups are needed at ingress, and MAC and IP lookups are performed at egress. As opposed to asymmetric, symmetric IRB implies the same number of lookups at ingress and egress.

  SR Linux support for symmetric IRB includes the prefix routing model using RT5s as in RFC 9136, including the interface-less IP-VRF-to-IP-VRF model (EVPN IFL model).

  SR Linux support for symmetric IRB also includes the advertisement and processing of host routes by using EVPN MAC/IP Advertisement routes, as specified in RFC 9135.

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

Compared to the asymmetric model, the symmetric model scales better because hosts' ARP and FDB entries are installed only on the directly attached leafs and not on all the leafs of the tenant.

The following sections show asymmetric and symmetric IFL forwarding configurations.

4.6.1.1 Asymmetric IRB

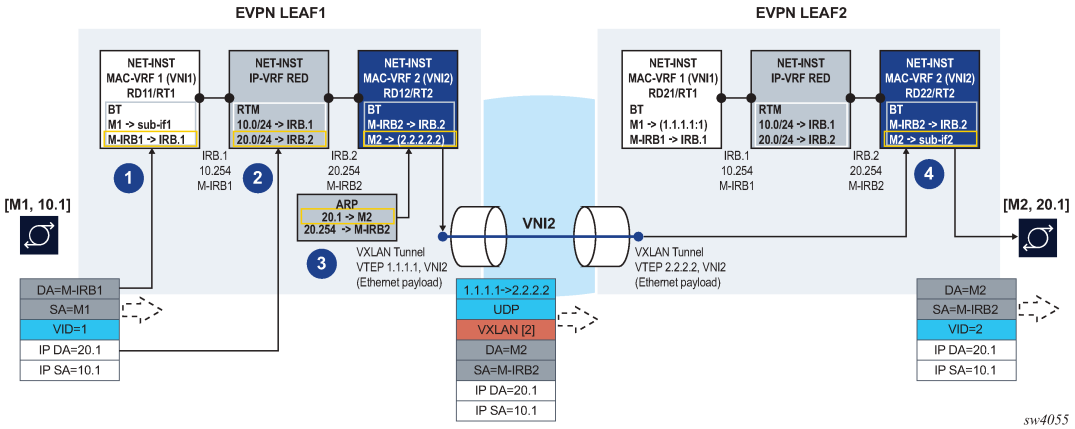
Note:
EVPN-VXLAN is supported on 7220 IXR-D2/D3/D2L/D3L, 7220 IXR-H6, 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

EVPN-MPLS is supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

The asymmetric IRB model is the basic Layer 3 forwarding model when the IP-VRF (or default network-instance) interfaces are all IRB-based. The asymmetric model assumes that all the subnets of a tenant are local in the IP-VRF/default route table, so there is no need to advertise EVPN RT5 routes.

The following figure shows the asymmetric IRB model for EVPN-VXLAN services. This model is also applicable to EVPN-MPLS services, where VXLAN encapsulation is replaced with MPLS encapsulation for Ethernet frames.

Figure 11: EVPN-VXLAN Layer 3 asymmetric forwarding



In this example, the host with IP address 10.1 (abbreviation of 10.0.0.1) sends a unicast packet with destination 20.1 (a host in a different subnet and remote leaf). Because the IP DA is in a remote subnet, the MAC DA is resolved to the local default gateway MAC, M-IRB1. The frame is classified for MAC lookup on MAC-VRF 1, and the result is IRB.1, which indicates that an IP DA lookup is required in IP-VRF red.

An IP DA longest-prefix match in the route table yields IRB.2, a local IRB interface, so an ARP and MAC DA lookup are required in the corresponding IRB interface and MAC-VRF bridge table.

The ARP lookup yields M2 on MAC-VRF 2, and the M2 lookup yields the EVPN destination [VTEP:VNI]=[2.2.2.2:2]. For EVPN-MPLS, this destination would be [TEP:label]. The routed packet is encapsulated with the corresponding inner MAC header and VXLAN or MPLS encapsulation before being sent to the wire.

In the asymmetric IRB model, if the ingress leaf routes the traffic via the IRB to a local subnet, and the destination MAC is aliased to multiple leaf nodes in the same ES destination, SR Linux can do load balancing on a per-flow basis.

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

EVPN Leaf 1 in Figure 11: EVPN-VXLAN Layer 3 asymmetric forwarding has the following configuration:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# info with-context
    interface ethernet-1/2 {
        admin-state enable
        vlan-tagging true
        subinterface 1 {
            type bridged
            admin-state enable
            vlan {
                encap {
                    single-tagged {
                        vlan-id 1
                    }
                }
            }
        }
    }
    interface irb0 {
        subinterface 1 {
            ipv4 {
                admin-state enable
                address 10.0.0.254/24 {
                    anycast-gw true
                }
            }
            anycast-gw {
            }
        }
        subinterface 2 {
            ipv4 {
                admin-state enable
                address 20.0.0.254/24 {
                    anycast-gw true
                }
            }
            anycast-gw {
            }
        }
    }
    network-instance ip-vrf-red {
        type ip-vrf
        interface irb0.1 {
        }
        interface irb0.2 {
        }
    }
    network-instance mac-vrf-1 {
        type mac-vrf
        admin-state enable
        interface ethernet-1/2.1 {
        }
        interface irb0.1 {
        }
        vxlan-interface vxlan1.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.1
                    evi 1
                }
```


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

```
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    network-instance mac-vrf-2 {
        type mac-vrf
        admin-state enable
        interface irb0.2 {
        }
        vxlan-interface vxlan1.2 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.2
                    evi 2
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    tunnel-interface vxlan1 {
        vxlan-interface 1 {
            type bridged
            ingress {
                vni 1
            }
        }
        vxlan-interface 2 {
            type bridged
            ingress {
                vni 2
            }
        }
    }
```

EVPN Leaf 2 in Figure 11: EVPN-VXLAN Layer 3 asymmetric forwarding has the following configuration:

```
--{ [FACTORY] + candidate shared default }--[  ]--
A:LEAF2# info with-context
    interface ethernet-1/12 {
        admin-state enable
        vlan-tagging true
        subinterface 1 {
            type bridged
            admin-state enable
            vlan {
                encap {
                    single-tagged {
                        vlan-id 2
                    }
                }
            }
        }
    }
    interface irb0 {
```


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

```
        subinterface 1 {
            ipv4 {
                admin-state enable
                address 10.0.0.254/24 {
                    anycast-gw true
                }
            }
            anycast-gw {
            }
        }
        subinterface 2 {
            ipv4 {
                admin-state enable
                address 20.0.0.254/24 {
                    anycast-gw true
                }
            }
            anycast-gw {
            }
        }
    }
    network-instance ip-vrf-red {
        type ip-vrf
        interface irb0.1 {
        }
        interface irb0.2 {
        }
    }
    network-instance mac-vrf-1 {
        type mac-vrf
        admin-state enable
        interface irb0.1 {
        }
        vxlan-interface vxlan1.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.1
                    evi 1
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    network-instance mac-vrf-2 {
        type mac-vrf
        admin-state enable
        interface irb0.2 {
        }
        vxlan-interface vxlan1.2 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.2
                    evi 2
                }
            }
```


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

```
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    tunnel-interface vxlan1 {
        vxlan-interface 1 {
            type bridged
            ingress {
                vni 1
            }
        }
        vxlan-interface 2 {
            type bridged
            ingress {
                vni 2
            }
        }
    }
```

When EVPN-MPLS is used instead of EVPN-VXLAN, the MAC-VRF network instances replace their VXLAN configuration with the configuration for MPLS resolution. For example, `mac-vrf-1` on Leaf 1 would have the following configuration:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# info
<snip>
    network-instance mac-vrf-1 {
        type mac-vrf
        admin-state enable
        interface ethernet-1/2.1 {
        }
        interface irb0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    evi 1
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [
                                ldp
                                sr-isis
                            ]
                        }
                    }
                }
            bgp-vpn {
            }
        }
    }
<snip>
```


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

4.6.1.2 Symmetric IRB interface-less ip-vrf-to-ip-vrf model

Note:
EVPN-VXLAN is supported on 7220 IXR-D2/D3/D2L/D3L, 7220 IXR-H5, 7220 IXR-H6, 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

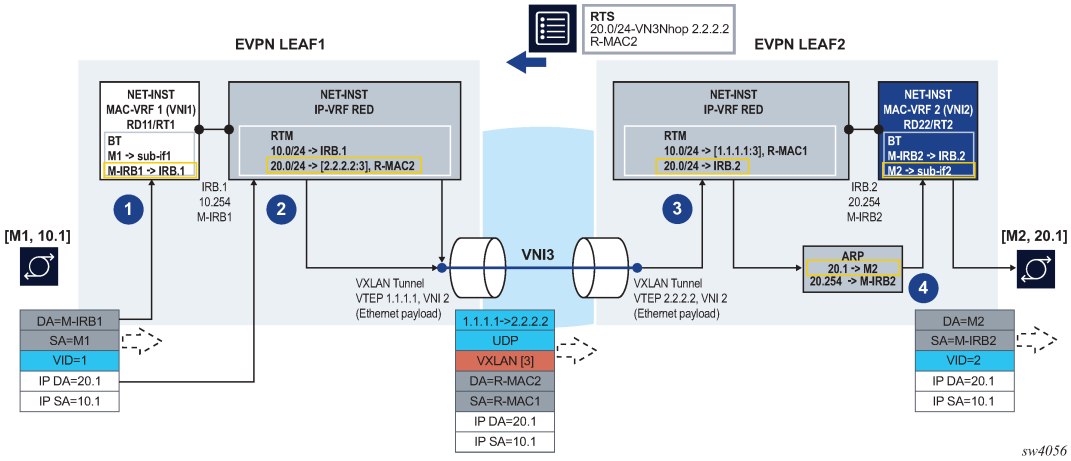
EVPN-MPLS is supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

SR Linux support for symmetric IRB is based on the prefix routing model using RT5s, and implements the EVPN interface-less (EVPN IFL) IP-VRF-to-IP-VRF model.

In the EVPN IFL model, all interface and local routes (static, ARP-ND, BGP, and so on) are automatically advertised in RT5s without the need for any export policy. Interface host and broadcast addresses are not advertised. On the ingress PE, RT5s are installed in the route table as indirect with owner BGP-EVPN.

The following figure shows the forwarding for symmetric IRB for EVPN-VXLAN services.

Figure 12: EVPN-VXLAN Layer 3 symmetric forwarding



As in the asymmetric model, the frame is classified for bridge-table lookup on the MAC-VRF and processed for routing in IP-VRF red.

In contrast to the asymmetric model, a longest prefix match does not yield a local subnet, but a remote subnet reachable via [VTEP:VNI]=[2.2.2.2:3] for EVPN-VXLAN and [TEP:label] for EVPN-MPLS, and inner MAC DA R-MAC2. SR Linux supports the EVPN interface-less (EVPN IFL) model, so that information is found in the IP-VRF route-table directly; a route lookup on the IP-VRF red route-table yields a VXLAN tunnel and VNI for EVPN-VXLAN and an MPLS tunnel and service label for EVPN-MPLS.

In the case of EVPN-VXLAN, as shown in Figure 12: EVPN-VXLAN Layer 3 symmetric forwarding:

- Packets are encapsulated with an inner Ethernet header and the VXLAN tunnel encapsulation.
- The inner Ethernet header uses the system-mac as MAC SA, and the MAC advertised along with the received RT5 as MAC DA. No VLAN tag is transmitted or received in this inner Ethernet header.

At the egress PE, the packet is classified for an IP lookup on the IP-VRF red (the inner Ethernet header is ignored).

The inner and outer IP headers are updated as follows:

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

- inner IP header TTL is decremented
- outer IP header TTL is set to 255
- outer DSCP value is marked as described in QoS for VXLAN tunnels
- no IP MTU check is performed before or after encapsulation

Because SR Linux supports EVPN IFL, the IP lookup in the IP-VRF red route-table yields a local IRB interface.

Subsequent ARP and MAC lookups provide the information to send the routed frame to subinterface 2.

EVPN Leaf 1 in Figure 12: EVPN-VXLAN Layer 3 symmetric forwarding has the following configuration:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# info with-context
    interface ethernet-1/2 {
        admin-state enable
        vlan-tagging true
        subinterface 1 {
            type bridged
            admin-state enable
            vlan {
                encap {
                    single-tagged {
                        vlan-id 1
                    }
                }
            }
        }
    }
    interface irb0 {
        subinterface 1 {
            ipv4 {
                admin-state enable
                address 10.0.0.254/24 {
                    anycast-gw true
                }
            }
            anycast-gw {
            }
        }
    }
    network-instance ip-vrf-red {
        type ip-vrf
        interface irb0.1 {
        }
        vxlan-interface vxlan1.3 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.3
                    evi 3
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    network-instance mac-vrf-1 {
```


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

```
        type mac-vrf
        admin-state enable
        interface ethernet-1/2.1 {
        }
        interface irb0.1 {
        }
        vxlan-interface vxlan1.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.1
                    evi 1
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    tunnel-interface vxlan1 {
        vxlan-interface 1 {
            type bridged
            ingress {
                vni 1
            }
        }
        vxlan-interface 3 {
            type routed
            ingress {
                vni 3
            }
        }
    }
```

EVPN Leaf 2 in Figure 12: EVPN-VXLAN Layer 3 symmetric forwarding has the following configuration:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# info with-context
    interface ethernet-1/12 {
        admin-state enable
        vlan-tagging true
        subinterface 2 {
            type bridged
            admin-state enable
            vlan {
                encap {
                    single-tagged {
                        vlan-id 2
                    }
                }
            }
        }
    }
    interface irb0 {
        subinterface 2 {
            ipv4 {
                admin-state enable
                address 20.0.0.254/24 {
                    anycast-gw true
                }
```


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

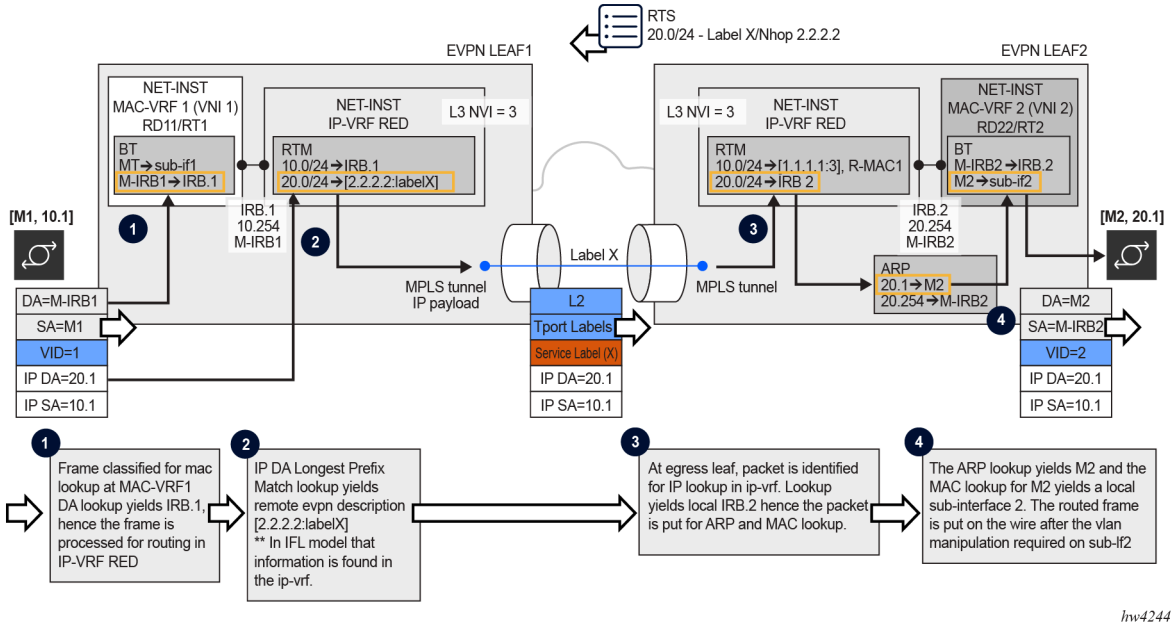
```
            }
            anycast-gw {
            }
        }
    }
    network-instance ip-vrf-red {
        type ip-vrf
        interface irb0.2 {
        }
        vxlan-interface vxlan1.3 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.3
                    evi 3
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    network-instance mac-vrf-2 {
        type mac-vrf
        admin-state enable
        interface ethernet-1/12.2 {
        }
        interface irb0.2 {
        }
        vxlan-interface vxlan1.2 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.2
                    evi 2
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    tunnel-interface vxlan1 {
        vxlan-interface 2 {
            type bridged
            ingress {
                vni 2
            }
        }
        vxlan-interface 3 {
            type routed
            ingress {
                vni 3
            }
        }
    }
```

In the case of EVPN-MPLS, MPLS tunnels exist between Leaf 1 and Leaf 2, instead of VXLAN.

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

The following figure shows the forwarding for symmetric IRB for EVPN-MPLS services.

Figure 13: EVPN-MPLS Layer 3 symmetric forwarding



The following characteristics apply in EVPN-MPLS encapsulation:

- To resolve EVPN-MPLS routes to tunnels in the tunnel table, the network instance is configured with **encapsulation-type mpls** and at least one MPLS tunnel type with the **bgp-evpn bgp-instance mpls next-hop-resolution allowed-tunnel-types [tunnel-types]** command.
- Received EVPN-MPLS next hops are resolved to the best tunnels in the allowed list.
- In order for MPLS to support the encapsulation of IP packets directly, packets are not encapsulated with an inner Ethernet header (as opposed to VXLAN which always requires inner Ethernet encapsulation).
- At the egress PE (shown as Leaf 2 in the preceding figure) and based on an ILM lookup on Label X, the packet is classified for an IP lookup on the `ip-vrf red` network instance.
- Subsequent ARP and MAC lookups provide the information required to send the routed packet to subinterface 2.

EVPN Leaf 1 in Figure 13: EVPN-MPLS Layer 3 symmetric forwarding has the following configuration.

```
--{ [FACTORY] + candidate shared default }--[ ]--
A:LEAF1# info with-context
    interface ethernet-1/2 {
        admin-state enable
        vlan-tagging true
        subinterface 1 {
            type bridged
            admin-state enable
            vlan {
                encap {
                    single-tagged {
                        vlan-id 1
                    }
                }
```


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

```
            }
        }
    }
    interface irb0 {
        subinterface 1 {
            ipv4 {
                address 10.0.0.254/24 {
                    anycast-gw true
                }
            }
            anycast-gw {
            }
        }
    }
    network-instance ip-vrf-red {
        type ip-vrf
        interface irb0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    encapsulation-type mpls
                    evi 3
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [ldp sr-isis]
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    network-instance mac-vrf-1 {
        type mac-vrf
        admin-state enable
        interface ethernet-1/2.1 {
        }
        interface irb0.1 {
        }
    }
```

EVPN Leaf 2 in Figure 13: EVPN-MPLS Layer 3 symmetric forwarding has the following configuration.

```
-{ [FACTORY] + candidate shared default }--[ ]--
A:LEAF2# info with-context
    interface ethernet-1/12 {
        admin-state enable
        vlan-tagging true
        subinterface 2 {
            type bridged
            admin-state enable
            vlan {
                encap {
                    single-tagged {
                        vlan-id 2
                    }
                }
            }
```


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

```
        }
    }
    interface irb0 {
        subinterface 2 {
            ipv4 {
                address 20.0.0.254/24 {
                    anycast-gw true
                }
            }
            anycast-gw {
            }
        }
    }
    network-instance ip-vrf-red {
        type ip-vrf
        interface irb0.2 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    encapsulation-type mpls
                    evi 3
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [ldp sr-isis]
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
    network-instance mac-vrf-2 {
        type mac-vrf
        admin-state enable
        interface ethernet-1/12.2 {
        }
        interface irb0.2 {
        }
    }
```

Note: Though it is not shown in this example, MAC-VRFs in Leaf 1 and Leaf 2 can also be EVPN-MPLS enabled.

4.6.1.3 Symmetric IRB model using EVPN MAC/IP advertisement routes in the route table

Note:
EVPN-VXLAN is supported on 7220 IXR-D2/D3/D2L/D3L, 7220 IXR-H5, 7220 IXR-H6, 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

EVPN-MPLS is supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

SR Linux supports the symmetric IRB model specified in RFC 9135. When enabled, this model allows a router to advertise the learned MAC and IP addresses of a host in an EVPN MAC/IP advertisement route

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

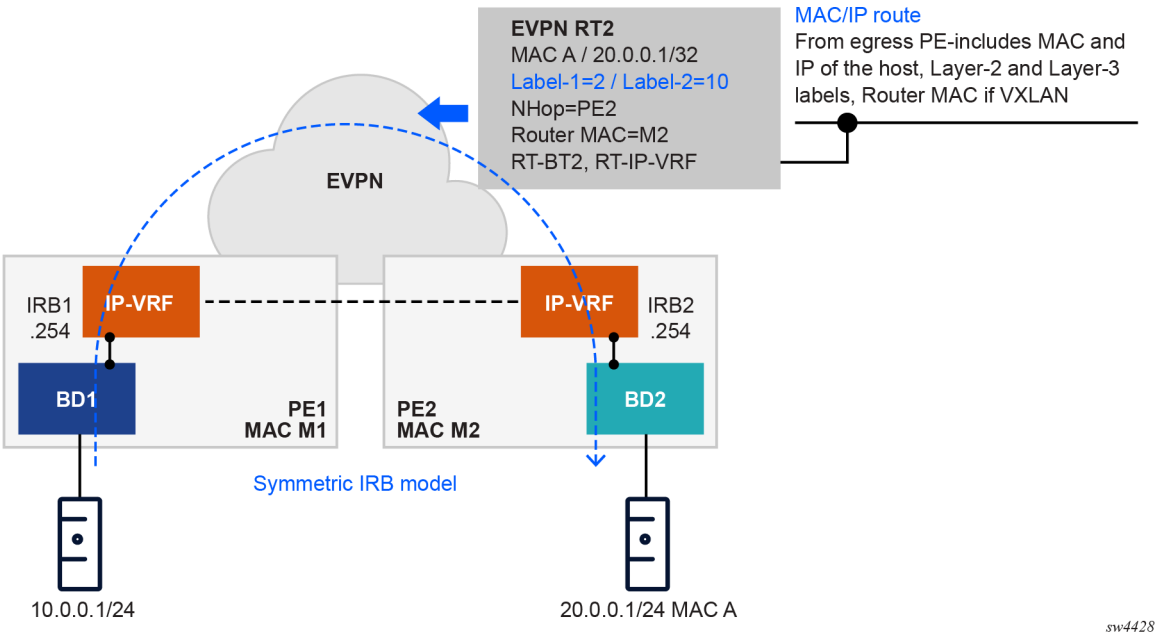
that can be imported into an IP-VRF. This route contains not only the label/VNI and route target of the MAC-VRF where the host is learned, but also the label/VNI and route target of the IP-VRF attached to the MAC-VRF network instance. This is possible because the MAC/IP advertisement route contains two labels/VNIs in the NLRI: label-1 (for Layer 2 purposes) and label-2 (for Layer 3 purposes).

On reception, the remote router receives the EVPN MAC/IP advertisement route and programs the IP address as a host route (/32 or /128) in the IP-VRF route table. Although host routes can be advertised in IP-prefix routes, the symmetric IRB model allows the advertisement of the MAC and host IP information in a single NLRI.

The host route in the IP-VRF route table has the new route type, **bgp-evpn-ifl-host**.

The following figure shows this symmetric IRB model.

Figure 14: Symmetric IRB model with EVPN MAC/IP advertisement



Example: Advertisement of EVPN IFL host routes

The advertisement behavior is enabled by the **interface-less-routing** command, as shown in the following example. The **interface-less-routing** command triggers the update of MAC/IP routes sent for the ARP/ND entries, including label-1, label-2, MAC-VRF, and IP-VRF route targets.

The **interface-less-routing** command determines if the EVPN MAC/IP advertisement routes sent for the ARP/ND entries carry the label-2 and IP-VRF route target information of the IP-VRF BGP-EVPN-IFL instance indicated by the **bgp-evpn-instance** command. The **bgp-evpn-instance** command can be set to values 1 (default) or 2.

```
--{ + candidate shared default }--[ interface irb0 subinterface 1 ipv4 arp evpn advertise
 dynamic ]--
# info detail
    internal-tags {
    }
    interface-less-routing {
        bgp-evpn-instance 1
```


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

```
    }

--{ + candidate shared default }--[  ]--
# info with-context network-instance IP-VRF-1
    network-instance IP-VRF-1 {
        type ip-vrf
        interface irb0.1 {
        }
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
                    evi 1
                    ecmp 8
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
```

The IP addresses received in MAC/IP advertisement routes with non-zero label-2 and the IP-VRF route target are programmed in the route table of the receiving router with route owner **bgp-evpn-ifl-host**, as follows.

```
--{ candidate shared default }--[  ]--
# info from state with-context network-instance IP-VRF-1 route-table ipv4-unicast route
 10.0.0.50/32 id 0 route-type * route-owner * origin-network-instance *
    network-instance IP-VRF-50 {
        route-table {
            ipv4-unicast {
                route 10.0.0.50/32 id 0 route-type bgp-evpn-ifl-host route-owner bgp_evpn_
ifl_host_mgr origin-network-instance IP-VRF-50 {
                    leakable false
                    metric 10
                    preference 170
                    active true
                    last-app-update "2025-02-13T11:06:12.889Z (a day ago)"
                    next-hop-group 25935383
                    next-hop-group-network-instance IP-VRF-50
                    resilient-hash false
                    fib-programming {
                        suppressed false
                        last-successful-operation-type add
                        last-successful-operation-timestamp "2025-02-13T11:06:12.892Z (a
 day ago)"
                        pending-operation-type none
                        last-failed-operation-type none
                    }
                }
            }
        }
    }
```


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

4.6.1.4 EVPN interface-ful model (EVPN IFF)

Note: This feature is supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, 7250 IXR Gen 3, and 7220 IXR-D*x* platforms.

The EVPN interface-ful (IFF) model allows you to interconnect multiple IP-VRFs using a Supplementary Broadcast Domain (SBD) as per RFC9136 sections 4.4.2 and 4.4.3. Unlike the EVPN IFL model where EVPN tunnels terminate on an IP-VRF, in the IFF model, the EVPN tunnels terminate on an SBD which is implemented as a MAC-VRF without bridged subinterfaces. The SBD connects to IP-VRFs through IRB subinterfaces.

SR Linux supports the following variations of the EVPN IFF model:

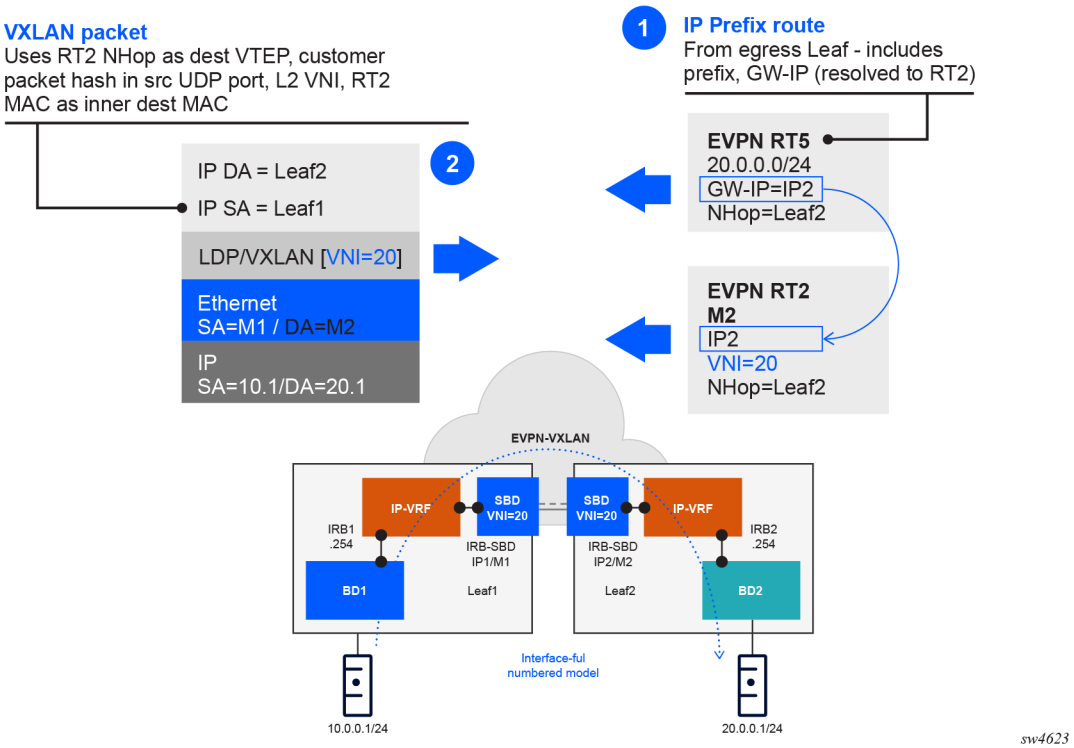
- IFF numbered model - as in RFC9136 section 4.4.3
- IFF unnumbered model - as in RFC9136 section 4.4.2

EVPN IFF numbered model

In the EVPN IFF numbered model, an IP prefix route uses the gateway IP address (GW-IP) field as the overlay index (next hop). The GW-IP is recursively resolved to the MAC/IP advertised route for that IP address, which is imported into the SBD.

The following figure shows an example of an EVPN IFF numbered model.

Figure 15: EVPN IFF numbered model



The model works as follows:

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

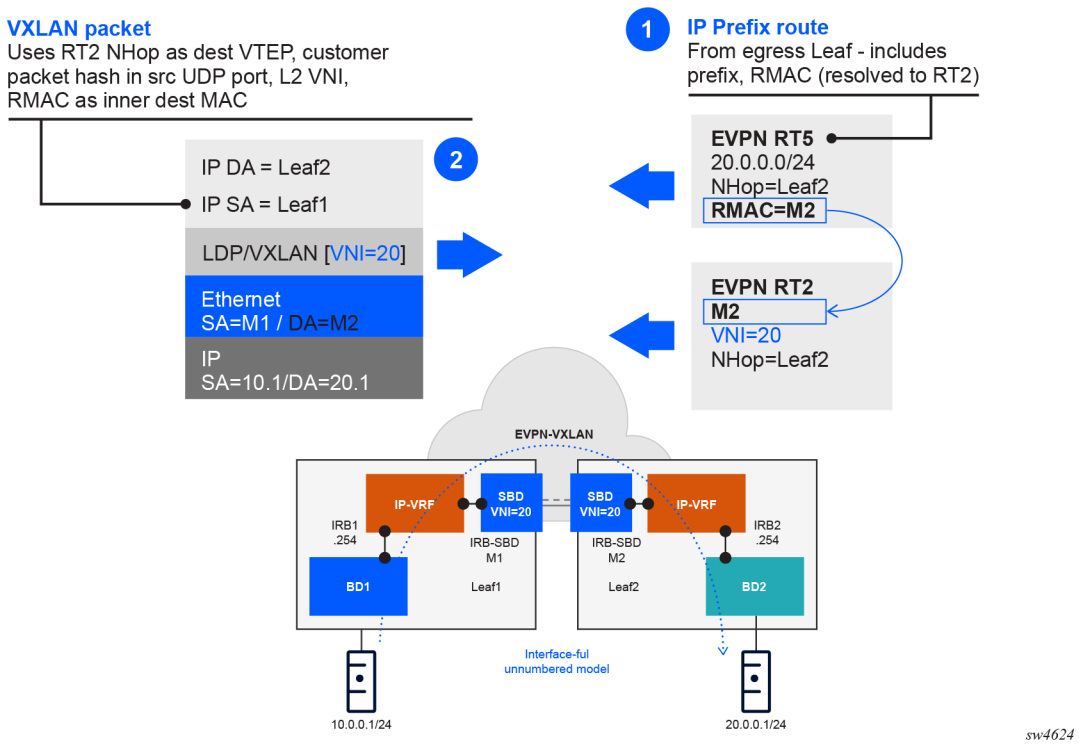
- The SBD interconnects the IP-VRFs.
- EVPN tunnels terminate in the SBD and use Layer 2 VNIs (in case of VXLAN).
- The GW-IP is assigned to the IRB interface of the SBD, as the primary IP address of the subinterface (IPv4 or IPv6).
- The MAC address and IP address of the IRB interface are advertised in an EVPN MAC/IP Advertisement route.
- In the preceding example, Leaf 1 resolves the IP prefix route recursively through the MAC/IP route using the GW-IP. This recursive resolution ensures faster convergence (multiple IP prefix routes can be invalidated with a single withdrawal of the overlay index (MAC/IP) route) and efficient mobility (multiple IP prefix routes can be advertised from an overlay index that can move or change dynamically).

EVPN IFF unnumbered model

The IFF unnumbered model is similar to the IFF numbered model except no IP address is configured on the SBD's IRB interface. Instead, the MAC address of the router is used as the overlay index (next hop).

The following figure shows an example of the EVPN IFF unnumbered model.

Figure 16: EVPN IFF unnumbered model



EVPN IFL and unnumbered IFF interoperability for VXLAN tunnels

The EVPN IFL and unnumbered EVPN IFF models are interoperable for VXLAN tunnels (and not for MPLS).

- Both use the same IP prefix route format and fields.

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

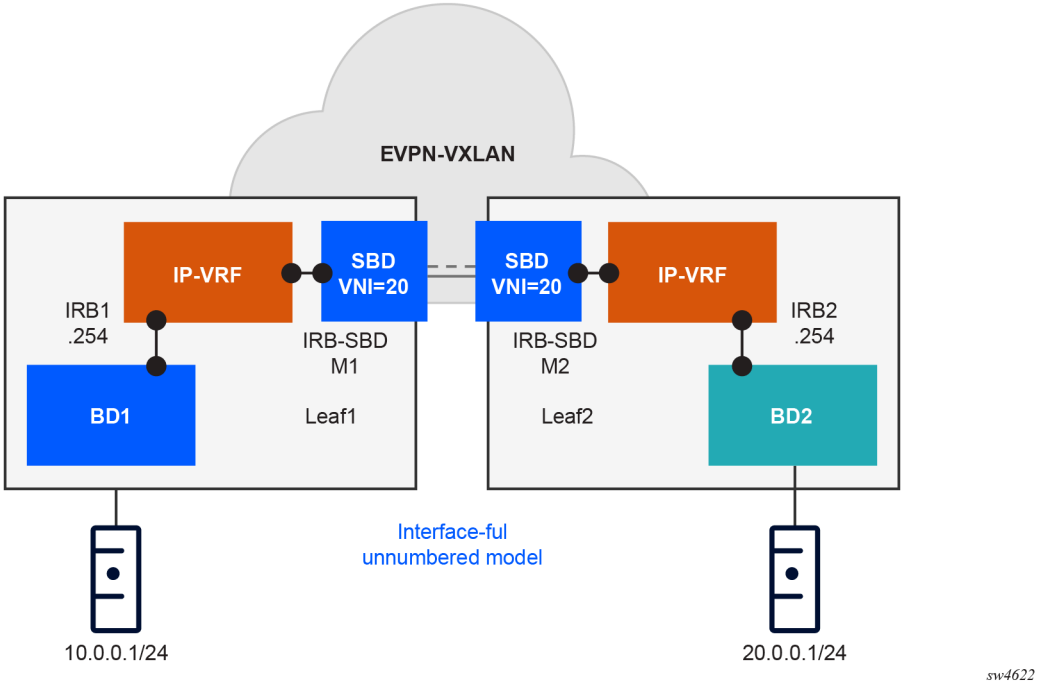
- In preceding example, a third leaf, Leaf 3 operating in EVPN IFL mode, can interwork with Leaf 2.
- In that case, Leaf 3 would not perform recursive resolution of the RT5 but uses the next hop of the RT5.
- When Leaf 1 or Leaf 2 operates in EVPN IFF unnumbered mode and receives an EVPN IFL route, they attempt to resolve the prefix using a MAC/IP advertisement route associated with the MAC address in the EVPN IFL route's MAC extended community. If no corresponding MAC/IP advertisement route is received, the leaf node generates an internal MAC/IP advertisement route using the MAC address from the extended community.

In contrast, the EVPN IFL and numbered EVPN IFF models are not interoperable.

EVPN IFF implementation

The following figure shows the configuration model for EVPN IFF for VXLAN transport. Although not illustrated in the figure, the EVPN-MPLS model follows the same implementation.

Figure 17: EVPN IFF implementation



In the preceding example, Leaf 2 has the following configuration:

- SBD configuration

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance SBD
    network-instance SBD {
        type mac-vrf
        interface irb0.0 {
            interface-ref {
                interface irb0
                subinterface 0
            }
        }
        vxlan-interface vxlan0.1 {
```


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

```
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
                    evi 1
                    supplementary-broadcast-domain {
                    }
                    routes {
                        route-table {
                            ip-prefix {
                                advertise-interface-ful true
                            }
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
```

- interface irb0 configuration - Numbered

```
--{ + candidate shared default }--[  ]--
# info with-context interface irb0
    interface irb0 {
        subinterface 0 {
            ipv4 {
                admin-state enable
                address 10.0.0.254/24 {
                    primary
                }
            }
        }
    }
```

- interface irb0 configuration - Unnumbered

```
--{ + candidate shared default }--[  ]--
# info with-context interface irb0
    interface irb0 {
        subinterface 0 {
            evpn-interface-ful-unnumbered {
            }
            ipv4 {
                admin-state enable
            }
        }
    }
```

- IP-VRF configuration

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance IP-VRF
    network-instance IP-VRF {
        type ip-vrf
        interface irb0.0 {
            interface-ref {
                interface irb0
```


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

```
                subinterface 0
            }
        }
        interface irb0.1 {
            interface-ref {
                interface irb0
                subinterface 1
            }
        }
    }
```

- BD2 and IRB configuration

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance BD2
    network-instance BD2 {
        type mac-vrf
        interface irb0.1 {
            interface-ref {
                interface irb0
                subinterface 1
            }
        }
        interface lag1.1 {
            interface-ref {
                interface lag1
                subinterface 1
            }
        }
        vxlan-interface vxlan0.2 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.2
                    evi 2
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
        }
    }
```

```
--{ + candidate shared default }--[  ]--
# info with-context interface irb0
    interface irb0 {
        subinterface 0 {
            ipv4 {
                admin-state enable
                address 10.0.0.254/24 {
                    primary
                }
            }
        }
        subinterface 1 {
            ipv4 {
                address 101.1.1.254/24 {
                    anycast-gw true
                }
                arp {
                    learn-unsolicited true
```


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

```
                    evpn {
                        advertise dynamic {
                        }
                    }
                }
            }
            anycast-gw {
            }
        }
    }
```

With the preceding configuration:

- VXLAN tunnels interconnecting both nodes are now created in the SBD, and not in the IP-VRF, as they were in the EVPN IFL case.
- The SBD must be configured with the commands, **network-instance protocols bgp-evpn bgp-instance supplementary-broadcast-domain** and **network-instance protocols bgp-evpn bgp-instance routes route-table ip-prefix advertise-interface-ful true**. The latter triggers the advertisement of EVPN IFF IP prefix routes.
- IP prefix routes and IRB MAC/IP routes are both processed and advertised in the context of the SBD. The format of the IP prefix routes follow RFC9136:
  - IP-VRF routes are advertised as EVPN IFF routes. The IP prefix routes carry the SBD's RD/RT, encapsulation and GW-IP address, if configured in the IRB.
    - If the **interface subinterface evpn-interface-ful-unnumbered** command is configured, the IP prefix route GW-IP is 0 and the router's MAC extended community encodes the SBD's IRB MAC. The same MAC address is advertised in the MAC/IP Advertisement route for the IRB.
    - If the **interface subinterface evpn-interface-ful-unnumbered** command is NOT configured, an IP address must be configured on the SBD's IRB. The value of the primary address is encoded in the GW-IP of the IP prefix routes advertised for the SBD, and in the IP address field of the MAC/IP Advertisement route advertised for the IRB, along with the IRB's MAC address.
  - When importing the routes, IP prefix routes are accepted into the EVPN IFF SBD only if their GW-IP matches the subnet of the IRB's primary IP address.

In the unnumbered configuration, IP prefix routes are recursively resolved to MAC/IP routes. These MAC/IP routes can be received from a peer or internally generated from the IP prefix route and its router's MAC extended community, if a MAC/IP route for the resolving MAC is not present in the SBD.

The configuration of the **interface subinterface evpn-interface-ful-unnumbered** command triggers the allocation of MAC next-hops that are used internally to link the received IP Prefix router's MAC address to the entry in the MAC-VRF. The MAC next-hops are limited resources and are monitored as follows:

```
--{ + candidate shared default }--[  ]--
# info from state platform linecard 1 forwarding-complex 0 datapath xdp resource mac-next-hops
    used-percent 0
    used-entries 1
    free-entries 10239
```

Where each received EVPN IFF unnumbered route with a unique router MAC address consumes one MAC next-hop.

An EVPN IFF route is displayed in the route table as a **bgp-evpn-iff** route as shown in the following example:

```
--{ + candidate shared default }--[  ]--
# show network-instance IP-VRF-50 route-table
```


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

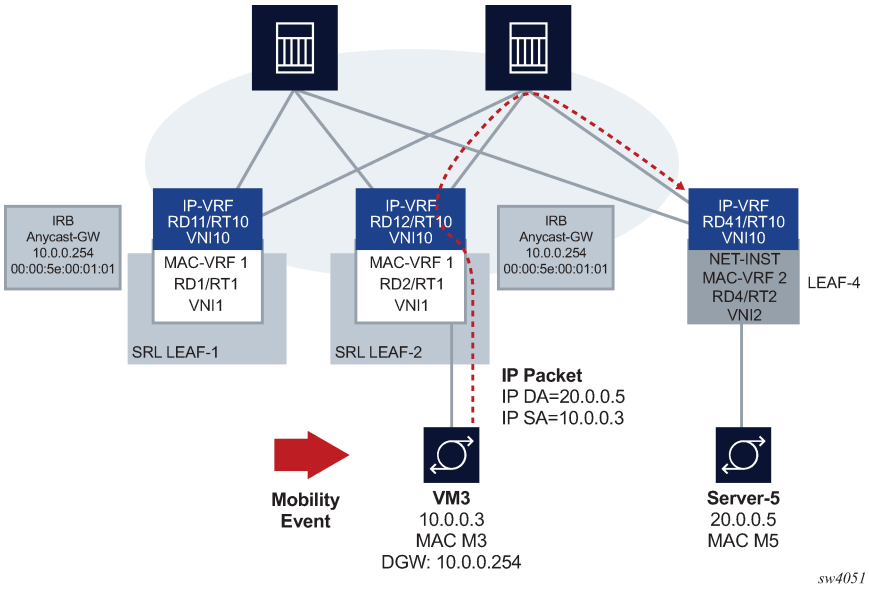
```
-------------------------------------------------------------------------------------------------
IPv4 unicast route table of network instance IP-VRF-50
-------------------------------------------------------------------------------------------------
+--------+------+------+------+------+--------+------+------+---------+------+--------+--------+
| Prefix |  ID  |Route |Route |Active|Origin  |Metric| Pref | Next-hop| Next |Backup  |Backup  |
|        |      |Type  |Owner |      |Network |      |      |  (Type) | -hop |Next-hop|Next-hop|
|        |      |      |      |      |Instance|      |      |         | Inter| (Type) | Inter  |
|        |      |      |      |      |        |      |      |         | face |        | face   |
|        |      |      |      |      |        |      |      |         |      |        |        |
|        |      |      |      |      |        |      |      |         |      |        |        |
|        |      |      |      |      |        |      |      |         |      |        |        |
+========+======+======+======+======+========+======+======+=========+======+========+========+
|3.3.3.3/| 0    | bgp- | evpn | True |IP-     | 0    | 169  | 1A:BF   | irb0 |        |        |
| 32     |      | evpn-| _mgr |      |VRF-50  |      |      |:04:FF   | .50  |        |        |
|        |      | iff  |      |      |        |      |      |:00:24   |      |        |        |
|        |      |      |      |      |        |      |      |interface|      |        |        |
|        |      |      |      |      |        |      |      |-with-mac|      |        |        |
+--------+------+------+------+------+--------+------+------+---------+------+--------+--------+
-------------------------------------------------------------------------------------------------
IPv4 routes total                    : 1
IPv4 prefixes with active routes     : 1
IPv4 prefixes with active ECMP routes: 0
-------------------------------------------------------------------------------------------------
```

4.6.2 Anycast gateways

Anycast gateways (anycast-GWs) are a common way to configure IRB subinterfaces in DC leaf nodes. Configuring anycast-GW IRB subinterfaces on all leaf nodes of the same BD avoids tromboning for upstream traffic from hosts moving between leaf nodes.

The following figure shows an example anycast-GW IRB configuration.

Figure 18: Anycast-GW IRB configuration



Anycast-GWs allow configuration of the same IP and MAC addresses on the IRB interfaces of all leaf switches attached to the same BD; for example, SRL LEAF-1 and SRL LEAF-2 in the preceding figure.

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

This feature optimizes the south-north forwarding because a host's default gateway always belongs to the connected leaf, irrespective of the host moving between leaf switches; for example VM3 moving from SRL LEAF-1 to SRL LEAF-2 in the figure.

When an IRB subinterface is configured as an anycast-GW, it must have one IP address configured as **anycast-gw**. The subinterface may or may not have other non-anycast-GW IP addresses configured.

To simplify provisioning, an option to automatically derive the anycast-GW MAC is supported, as described in RFC 9135. The auto-derivation uses a virtual-router-id similar to MAC auto-derivation in RFC 5798 (VRRP). Anycast-GWs use a default virtual-router-id of 01, if not explicitly configured. Because only one anycast-gw-mac per IRB sub-interface is supported, the anycast-gw-mac for IPv4 and IPv6 is the same in the IRB sub-interface.

The following is an example configuration for an anycast-GW subinterface:

```
// Configuration Example of an anycast-gw IRB sub-interface

A:leaf-1/2# info with-context interface irb1 subinterface 1
  interface irb1 {
    subinterface 1 {
      ipv4 {
        admin-state enable
        address 10.0.0.254/24 {
          primary true
          anycast-gw true
        }
      }
      anycast-gw {
        virtual-router-id 2
      }
    }
  }

// State Example of an anycast-gw IRB sub-interface


A:leaf-1/2# info from state with-context
  ipv4 {
    address 10.0.0.254/24 {
      primary true
      anycast-gw true
    }
  }
  anycast-gw {
    virtual-router-id 2
    anycast-gw-mac 00:00:5e:00:01:02
    anycast-gw-mac-origin auto-derived
  }
```

The **anycast-gw true** command designates the associated IP address as an anycast-GW address of the subinterface and associates the IP address with the **anycast-gw-mac** address in the same sub-interface. ARP requests or NSs received for the anycast-GW IP address are replied using the **anycast-gw-mac** address, as opposed to the regular system-derived MAC address. Similarly, CPM-originated GARPs or unsolicited NAs sent for the anycast-GW IP address use the **anycast-gw-mac** address as well. Packets routed to the IRB use the **anycast-gw-mac** as the SA in Ethernet header.

All IP addresses of the IRB subinterface and their associated MACs are advertised in MAC/IP routes with the static flag set. The non-anycast-GW IPs are advertised along with the interface hardware MAC address, and the anycast-GW IP addresses along with the anycast-gw-mac address.

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

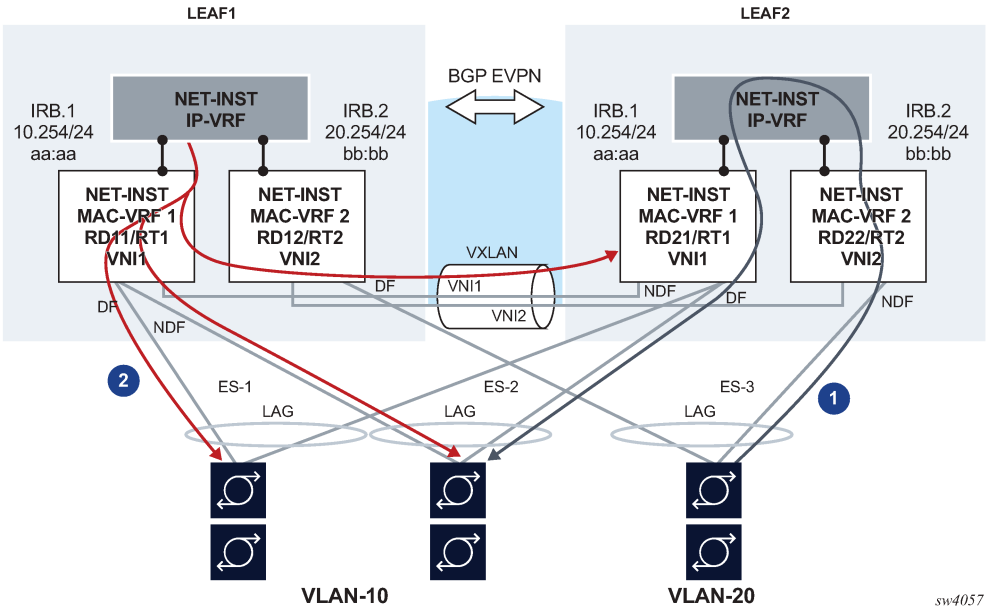
In addition, the **anycast-gw true** command makes the system skip the ARP/ND duplicate-address-detection procedures for the anycast-GW IP address.

4.6.3 EVPN Layer 3 multihoming and anycast gateways

In an EVPN Layer 3 scenario, all IRB interfaces facing the hosts must have the same IP address and MAC; that is, an anycast-GW configuration. This avoids inefficiencies for all-active multihoming or speeds up convergence for host mobility.

The use of anycast-GW along with all-active multihoming in an EVPN-VXLAN scenario is shown in the following figure:

Figure 19: EVPN-VXLAN Layer 3 model – multihoming and anycast gateway



In this example:

1. Routed unicast traffic is always routed to the directly connected leaf (no tromboning).
2. BUM traffic sent from the IRB interface is sent to all DF and NDF subinterfaces, similar to BUM entering a subinterface.

   This applies to:

   - system-generated unknown unicast or broadcast traffic
   - ARP requests, GARPs, and Neighbor Discovery multicast messages
   - unicast traffic with unknown MAC DA

This solution also applies when EVPN-MPLS is used on MAC-VRFs instead of VXLAN. Flooded traffic coming from the IRB interface to the MAC-VRF is flooded to local DF subinterfaces and to other leaf nodes attached to the same broadcast domain.

When a host connected to ES-3 sends a unicast flow to be routed in the IP-VRF, the flow must be routed in the leaf receiving the traffic, irrespective of the server hashing the flow to Leaf-1 or Leaf-2. To do this, the host is configured with only one default gateway, 20.254/24. When the host ARPs for it, it does not matter if

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

the ARP request is sent to Leaf-1 or Leaf-2. Either leaf replies with the same anycast-GW MAC, and when receiving the traffic either leaf can route the packet.

This scenario is supported on IP-VRF network instances and the default network instance.

Note:
When IRB subinterfaces are attached to MAC-VRF network instances with all-active multihoming ESs, set the `arp timeout` / `neighbor-discovery stale-time` settings on the IRB subinterface to a value that is 30 seconds lower than the `age-time` configured in the MAC-VRF. This setting avoids transient packet loss situations triggered by the MAC address of an active ARP/ND entry being removed from the MAC table.

4.6.4 EVPN Proxy MAC/IP Advertisement routes in MAC-VRFs with IRB interfaces

Note: This feature is supported on 7220 IXR and 7250 IXR (excluding mixed chassis) platforms.

Node failures in EVPN multi-homing (MH) environments cause significant control plane churn. In EVPN MH with IRB scenarios, this manifests as delayed downstream convergence because of large-scale ARP re-resolution.

Proxy MAC/IP advertisement on IRB interfaces accelerates convergence after a node failure in EVPN MH scenarios. The remaining nodes reoriginate and retain the MAC/IP Advertisement routes that were originally advertised by the failed node.

To configure this behavior, use the **local-es entry-type evpn-local-es** command, as follows:

- **local-es**

  This container provides the context for the configuration of EVPN local Ethernet segements for the ARP/ND entries.

- **entry-type**

  The **entry-type** configuration determines if the ARP/ND entry created for EVPN RT2 routes with the ESI matching a local ESI is installed as type **evpn** or type **evpn-local-es**. If the associated EVPN route that created the evpn-local-es entry changes ESI to 0, the re-originated entry transitions back to a standard EVPN type.

- **retention-timer**

  The retention timer is specified in seconds, with a value in the range of 0 to 65535 and a default value of 0. To configure it, the **entry-type** must be configured as **evpn-local-es**. The timer starts when the EVPN route that created the **evpn-local-es** entry is withdrawn. If the **evpn-local-es** entry is not refreshed before the timeout expires, it is deleted and the associated routes are withdrawn.

Configuring **local-es entry-type evpn-local-es** has the following effects:

- ARP/ND entries that are received as EVPN are re-originated, if their corresponding EVPN MAC/IP routes contain an ESI that matches that of a local ES in the service, and the **advertise evpn-local-es** command is configured.
- The resulting entries are programmed as the new type **evpn-local-es**.
- The new type of entry is displayed as **evpn-local-es** in the ARP/ND table and triggers the advertisement of local EVPN MAC/IP routes with the local service information and the proxy bit set.
- If **advertise evpn-local-es interface-less-routing** is enabled, the PE re-originates and re-advertises the received **evpn-local-es** entries as EVPN-IFL-HOST routes with P=1 (Proxy flag encoded in the

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

ARP/ND extended community of the MAC/IP Advertisement route). The remote PE applies IP aliasing in the IP-VRF for non-zero ESI EVPN-IFL-HOST routes.

- If **host-route populate evpn-local-es** is enabled, the PE also creates local ARP/ND host routes for the **evpn-local-es** entries.

The following examples shows how this feature is configured on an IRB subinterface attached to a MAC-VRF with Ethernet Segments. All multi-homed leaf nodes (leaf1 and leaf2 in this example) must use the same configuration on their IRB subinterfaces.

Example: ARP configuration

The example shows the ARP configuration; the neighbor discovery configuration for IPv6 is equivalent.

```
--{ * candidate shared default }--[  ]--
A:leaf1# info with-context interface irb0
    interface irb0 {
        subinterface 12 {
            ipv4 {
                admin-state enable
                address 10.0.2.254/24 {
                    anycast-gw true
                    primary
                }
                arp {
                    learn-unsolicited true
                    debug [
                        messages
                    ]
                    evpn {
                        advertise dynamic {
                            interface-less-routing {
                                bgp-evpn-instance 1
                            }
                        }
                        advertise evpn-local-es {
                            interface-less-routing {
                                bgp-evpn-instance 1
                            }
                        }
                        local-es {
                            entry-type evpn-local-es
                            retention-timer 100
                        }
                    }
                }
            }
            anycast-gw {
            }
        }
    }

--{ * candidate shared default }--[  ]--
A:leaf1# info with-context system network-instance protocols evpn
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment s12 {
                                admin-state enable
```


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

```
                                esi 01:12:00:00:00:00:00:00:00:00
                                interface ethernet-1/2 {
                                }
                                advertise-ifl-host-ad-routes {
                                    bgp-evpn-instance 1
                                }
                            }
                        }
                    }
                }
            }
        }
    }

--{ * candidate shared default }--[  ]--
A:leaf1# info with-context network-instance IP-VRF-1
    network-instance IP-VRF-1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
                    evi 1
                    ecmp 2
                }
            }
        }
    }
```

Example: ATP tables

```
A:leaf1# show arpnd arp-entries interface irb0 subinterface 12
+----------+----------+-------------+---------+-------------+------------------+-------+
| Network  |          |             |         |             |                  |       |
| instance | Interface| Subinterface| Neighbor|    Origin   | Link layer       |       |
|          |          |             |         |             | address          | Expiry|
+==========+==========+=============+=========+=============+==================+=======+
| IP-VRF-1, | irb0     |           12| 10.0.2.1|          evpn| AA:C1:AB:FF:B3:50|       |
| MAC-VRF-12|          |             |         |             |                  |       |
|          |          |             |         |             |                  |       |
| IP-VRF-1, | irb0     |           12| 10.0.2.2| evpn-local-es| AA:C1:AB:91:C5:FC|       |
| MAC-VRF-12|          |             |         |             |                  |       |
+----------+----------+-------------+---------+-------------+------------------+-------+
  Network instance : all
  Total entries : 2 (0 static, 1 evpn, 1 evpn-local-es, 0 dynamic)

A:leaf2# show arpnd arp-entries interface irb0 subinterface 12
+----------+----------+-------------+---------+--------+------------------+----------+
| Network  |          |             |         |        |                  |          |
| instance | Interface| Subinterface| Neighbor| Origin | Link layer       |          |
|          |          |             |         |        | address          |  Expiry  |
+==========+==========+=============+=========+========+==================+==========+
| IP-VRF-1, | irb0     |           12| 10.0.2.1| evpn   | AA:C1:AB:FF:B3:50 |          |
| MAC-VRF-12|          |             |         |        |                  |          |
|          |          |             |         |        |                  |          |
| IP-VRF-1, | irb0     |           12| 10.0.2.2| dynamic| AA:C1:AB:91:C5:FC |          |
| MAC-VRF-12|          |             |         |        |                  | 3 hours  |
|          |          |             |         |        |                  | from now |
+----------+----------+-------------+---------+--------+------------------+----------+
  Network instance : all
  Total entries : 2 (0 static, 1 evpn, 0 evpn-local-es, 1 dynamic)
```


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

Example: Proxy MAC/IP Advertisement routes on remote leaves

```
A:leaf3# show network-instance default protocols bgp routes evpn route-type 2 ip-address
 10.0.2.2 detail   (best paths)
Route Distinguisher: 100.0.0.1:12
    ESI                   : 01:12:00:00:00:00:00:00:00:00
    Label                 : 12+1
    BGP next-hop          : 100.0.0.1
    Communities           : [target:64500:12, target:64500:1, arp-nd:R:0/O:0/P:1/I:0, mac-
nh:1a:da:02:ff:00:00, bgp-tunnel-encap:VXLAN]
Route Distinguisher: 100.0.0.2:12
    ESI                   : 01:12:00:00:00:00:00:00:00:00
    Label                 : 12+1
    BGP next-hop          : 100.0.0.2
    Communities           : [target:64500:12, target:64500:1, mac-nh:1a:10:03:ff:00:00,
 bgp-tunnel-encap:VXLAN]
```

4.6.5 EVPN Layer 3 host route mobility

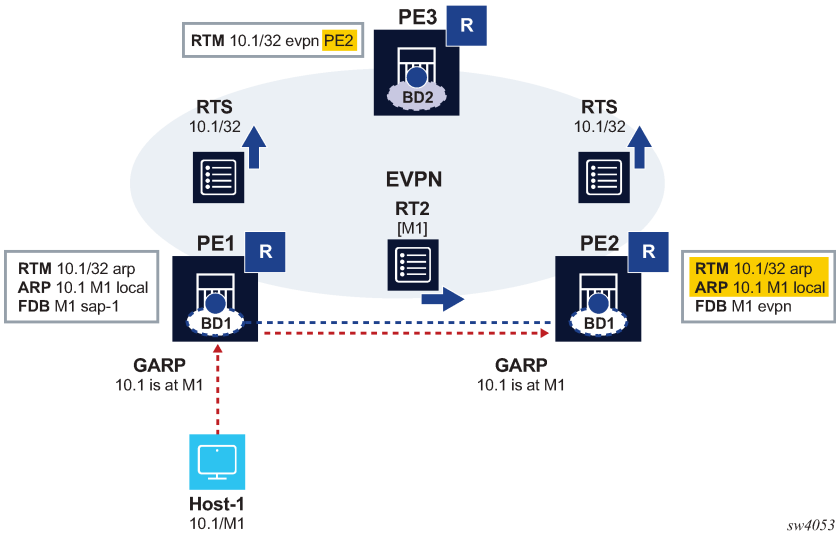
EVPN host route mobility refers to the procedures that allow the following:

- learning ARP/ND entries out of unsolicited messages from hosts
- generating host routes out of those ARP/ND entries
- refreshing the entries when mobility events occur within the same BD

EVPN host route mobility is part of basic EVPN Layer 3 functionality as defined in RFC 9135.

The following figure EVPN host route mobility.

Figure 20: EVPN host route mobility



In Figure 20: EVPN host route mobility a host is attached to PE1, so all traffic from PE3 to that host must be forwarded directly to PE1. When the host moves to PE2, all the tables must be immediately updated so that PE3 sends the traffic to PE2. EVPN host route mobility works by doing the following:

1. snooping and learning ARP/ND entries for hosts upon receiving unsolicited GARP or NA messages

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

2. creating host routes out of those dynamic ARP/ND entries. These routes only exist in the control plane and are not installed in the forwarding plane to avoid FIB exhaustion with too many /32 or /128 host routes.
3. advertising the locally learned ARP/ND entries in MAC/IP routes so that ARP/ND caches can be synchronized across leaf nodes
4. advertising the host routes as IP prefix routes
5. triggering ARP/ND refresh messages for changes in the ARP/ND table or MAC table for a specific IP, which allows updating the tables without depending on the ARP/ND aging timers (which can be hours long)

The following configuration enables an anycast-GW IRB subinterface to support mobility procedures:

```
// Example of the configuration of host route mobility features

--{ candidate shared default }--[  ]--
A:-# info with-context interface irb1 subinterface 1
    interface irb1 {
        subinterface 1 {
            ipv4 {
                admin-state enable
                address 10.0.0.254/24 {
                    anycast-gw true
                    primary
                }
                arp {
                    learn-unsolicited true
                    host-route {
                        populate static
                        populate dynamic
                    }
                    evpn {
                        advertise static
                        advertise dynamic
                    }
                }
            }
            ipv6 {
                admin-state enable
                address 200::254/64 {
                    anycast-gw true
                    primary
                }
                neighbor-discovery {
                    learn-unsolicited true
                    host-route {
                        populate static
                        populate dynamic
                    }
                    evpn {
                        advertise static
                        advertise dynamic
                    }
                }
            }
            anycast-gw {
            }
        }
    }
--{ candidate shared default }--[  ]--
# info from state with-context
```


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

```
    admin-state enable
    ip-mtu 1500
    name irb1.1
    ifindex 1082146818
    oper-state up
    last-change "a minute ago"
    ipv4 {
        allow-directed-broadcast false
        address 10.0.0.254/24 {
            anycast-gw true
            origin static
        }
        arp {
            duplicate-address-detection true
            timeout 14400
            learn-unsolicited true
            host-route {
                populate static
                populate dynamic
                }
            }
        }
    }
    ipv6 {
        address 200::254/64 {
            anycast-gw true
            origin static
            status unknown
        }
        address fe80::201:1ff:feff:42/64 {
            origin link-layer
            status unknown
        }
        neighbor-discovery {
            duplicate-address-detection true
            reachable-time 30
            stale-time 14400
            learn-unsolicited both
            host-route {
                populate static
                populate dynamic
            }
        }
        router-advertisement {
            router-role {
                current-hop-limit 64
                managed-configuration-flag false
                other-configuration-flag false
                max-advertisement-interval 600
                min-advertisement-interval 200
                reachable-time 0
                retransmit-time 0
                router-lifetime 1800
            }
        }
    }
    anycast-gw {
        virtual-router-id 1
        anycast-gw-mac 00:00:5E:00:01:01
        anycast-gw-mac-origin vrid-auto-derived
    }
...
```


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

In this configuration, when `learn-unsolicited` is set to `true`, the node processes all solicited and unsolicited ARP/ND flooded messages received on subinterfaces (no VXLAN/MPLS) and learns the corresponding ARP/ND entries as dynamic. By default, this setting is `false`, so only solicited entries are learned by default.

The advertisement of EVPN MAC/IP routes for the learned ARP entries must be enabled/disabled by configuration; it is disabled by default. In the example above, this is configured with the `advertise dynamic` and `advertise static` settings.

The creation of host routes in the IP-VRF route table out of the dynamic or static ARP entries can be enabled/disabled by configuration; it is disabled by default. In the example above, this is configured with the `host-route populate dynamic` and `host-route populate static` settings.

By default, the ARP/ND host routes are not installed in the data path. You can configure `host-route populate dynamic datapath-programming true` to change this default behavior so that the ARP/ND host routes are installed in the data path. This configuration is required when the ARP/ND host routes are leaked. See "Network instance route leaking" in the *SR Linux Configuration Basics Guide*.

The dynamic ARP entries are refreshed without any extra configuration. The system sends ARP requests for the dynamic entries to make sure the hosts are still alive and connected.

4.6.6 EVPN-VXLAN IFL interoperability with EVPN IFF

By default, the SR Linux EVPN IFL (interface-less) model, as defined in Symmetric IRB interface-less ip-vrf-to-ip-vrf model for VXLAN, does not interoperate with the EVPN IFF (interface-ful) model implemented by third-party vendors. However, it is interoperable with the SR Linux EVPN IFF unnumbered model, because the latter can internally derive a MAC/IP advertisement route from the MAC extended community of the IP prefix route encoded in the EVPN IFF unnumbered format. Additionally, the SR Linux EVPN IFL model can be configured to interoperate with third-party EVPN IFF unnumbered implementations that require explicit generation of a MAC/IP advertisement route for the MAC next hop.

To do this, configure the **advertise-gateway-mac** command for the IP-VRF network instance. When this command is configured, the node advertises a MAC/IP route using the following:

- gateway-mac for the IP-VRF (that is, the system-mac)
- RD/RT, next-hop, and VNI of the IP-VRF where the command is configured
- null IP address, ESI or Ethernet Tag ID

Some devices support two EVPN Layer 3 IPv6 modes: IFF unnumbered and IFF numbered, as described in RFC 9136. The SR Linux interoperability mode enabled by the **advertise-gateway-mac** command only works with devices that use the EVPN IFF unnumbered model. This is because the EVPN IFL and EVPN IFF unnumbered models both use the same format in the IP prefix route, and they differ only in the additional MAC/IP route for the gateway-mac. The EVPN IFL and EVPN IFF numbered models have different IP prefix route formats, so they cannot interoperate.

The following example enables interoperability with the EVPN IFF unnumbered model on EVPN IFL instances

:

```
--{ [FACTORY] + candidate shared default }--[  ]--
# info from state with-context network-instance protocols
  bgp-evpn {
      bgp-instance 1 {
          admin-state enable
```


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

```
        vxlan-interface vxlan1.2
        routes {
            route-table {
                mac-ip {
                    advertise-gateway-mac true
                }
            }
        }
    }
}
```

4.6.7 VIP discovery for redundant servers

In data centers, it is common to have clusters of servers sharing the same IP address and working in an active-standby mode, so that only one of the servers in the cluster is active at a time. This shared IP address is known as a virtual IP (VIP) address. SR Linux can discover which server in the cluster owns the VIP address.

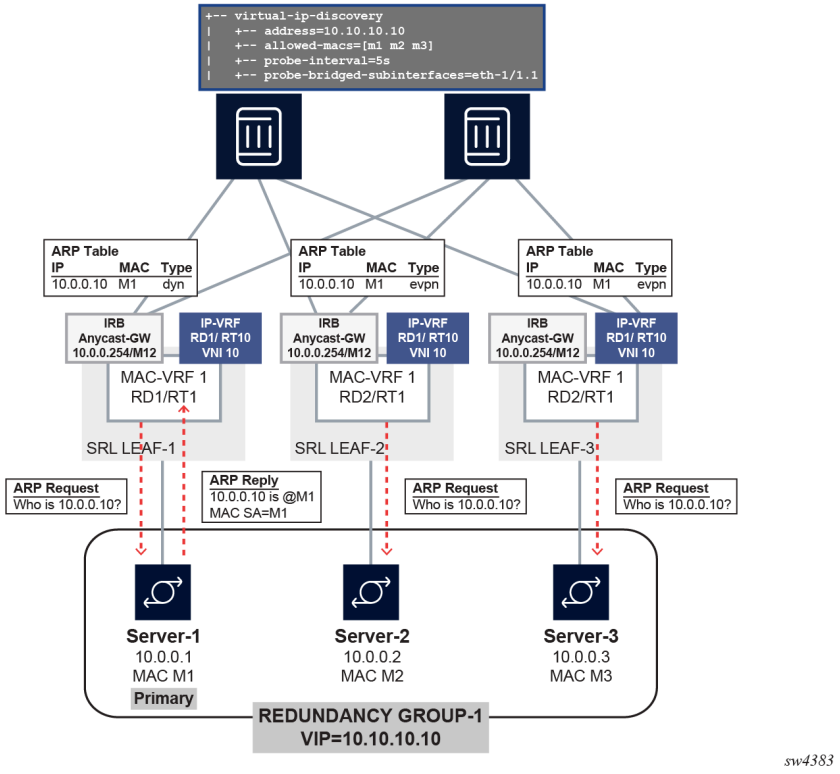
ARP requests (NS for IPv6) from a leaf node or gratuitous ARP (unsolicited NA for IPv6) from a server are used to discover which host owns the VIP. Among the servers sharing the VIP, only the active one either sends a gratuitous ARP (or unsolicited NA) or replies to the ARP request (NS) from the leaf node. If a server does not send a gratuitous ARP, you can optionally configure SR Linux to send ARP requests periodically at a specified probe interval.

The leaf nodes create entries in their ARP tables that map the VIP to the MAC address of the active server. You can optionally configure a list of allowed MAC addresses so that the ARP/ND entry for the VIP is created only if the reply from the server comes from one of the MACs on the allowed list.

The following figure shows a configuration where three SR Linux devices in a MAC-VRF are connected to a redundant server group.

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

Figure 21: VIP discovery for the active server in a server group



In this configuration, the SR Linux devices (SRL LEAF-1/2/3) start sending ARP requests for the VIP 10.0.0.10 on the ethernet 1/1.1 subinterface every 5 seconds until the ARP entry for the VIP/VMAC is learned. The ARP entry is created only if the MAC address matches one of the allowed MACs: M1, M2, or M3.

Initially, Server-1 is the primary server in the server group, and it responds to the ARP request from Leaf-1, causing the server's MAC address M1 to be added to the ARP table on the SR Linux devices for VIP 10.0.0.10. If Server-2 (MAC M2) becomes the primary server, it sends out a gratuitous ARP message or responds to an ARP request from LEAF-2, which triggers an update of the ARP table on Leaf-2 and a MAC/IP route update with 10.0.0.10/M2. When Leaf-1 receives the update, it updates its ARP table immediately with the new entry.

VIP discovery is supported on single-homed and multihomed subinterfaces. For all-active multihoming, and based on the local-bias behavior, the ARP/ND probes are sent to all ES subinterfaces irrespective of their DF state. When the active server replies, the MAC/IP route is synchronized in the other ES peers based on the functionality described in EVPN Layer 3 host route mobility.

4.6.7.1 Configuring VIP discovery

Procedure

To configure VIP discovery for an IRB interface, you specify the bridged subinterfaces that can be sent probe messages (ARP requests) for the VIP, optionally a probe interval that determines how often the

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

probe messages are sent, and optionally a list of allowed MAC addresses that indicate the MAC addresses of the servers in the group.

Example: Configure VIP discovery for IPv4

```
--{ candidate shared default }--[  ]--
# info with-context interface irb1 subinterface 1 ipv4 arp
 interface irb1 {
        subinterface 1 {
            ipv4 {
                admin-state enable
                arp {
                    virtual-ipv4-discovery {
                        address 10.10.10.10 {
                            probe-bridged-subinterfaces [
                                ethernet-1/1.1
                            ]
                            probe-interval 5
                            allowed-macs [
                                00:14:9C:78:E2:E1
                                00:14:9C:78:E2:E2
                                00:14:9C:78:E2:E3
                            ]
                        }
                    }
                }
            }
        }
    }
```

Example: Configure VIP discovery for IPv6

```
--{ candidate shared default }--[  ]--
# info with-context interface irb1 subinterface 1 ipv6 neighbor-discovery
    interface irb1 {
        subinterface 1 {
            ipv6 {
                admin-state enable
                neighbor-discovery {
                    virtual-ipv6-discovery {
                        address 2001::10:10:10:10 {
                            probe-interval 5
                            allowed-macs [
                                00:14:9C:78:E2:E1
                                00:14:9C:78:E2:E2
                                00:14:9C:78:E2:E3
                            ]
                            probe-bridged-subinterfaces [
                                ethernet-1/1.1
                            ]
                        }
                    }
                }
            }
        }
    }
```

In this configuration, SR Linux starts sending ARP requests for VIP 10.10.10.10 (**address** parameter) on the `ethernet 1/1.1` subinterface (**probe-bridged-subinterfaces** parameter) every 5 seconds (**probe-interval** parameter) so that the ARP entry for the VIP/VMAC is learned. The ARP entry

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

is created only if the MAC address matches one of the MACs in the allowed list (**allowed-macs** parameter).

Up to 10 subinterfaces can be specified in the **probe-bridged-subinterfaces** list. The probe messages are sent only to the subinterfaces in this list. If no subinterfaces are specified in the list, no probe messages are sent.

The default **probe-interval** is 0, meaning that periodic probe messages are not sent by default. However regardless of the **probe-interval** setting, a probe message is sent to the subinterfaces in the **probe-bridged-subinterfaces** list whenever the following occur:

- the VIP is configured
- when the IRB interface becomes operationally up (when the prefix becomes preferred)
- when the ARP/ND entry corresponding to the VIP moves from type dynamic to type EVPN, or from type dynamic to type dynamic (changing to a different MAC)
- when subinterfaces in the **probe-bridged-subinterfaces** list become operationally up
- when the arpnd_mgr application restarts
- when the **allowed-macs** list, **probe-interval**, or **probe-bridged-subinterfaces** list changes

When the **probe-interval** is set to a non-zero value, SR Linux keeps probing for the VIP continuously, even after the ARP/ND entry for the VIP is created.

Up to 10 MAC addresses can be specified in the **allowed-macs** list. The ARP/ND entry for the VIP is created only if the resolving MAC address corresponds to one of the MAC addresses in the list. If no MAC addresses are specified in the list, any resolving MAC is valid for the creation of the ARP/ND entry. When a MAC address included in the **allowed-macs** list is used in an existing ARP/ND entry, and the MAC is removed from the list, the ARP/ND entry is deleted.

4.6.7.2 Displaying VIP discovery information

Procedure

You can display the number of probe packets sent by the SR Linux device using the **info from state** command. Statistics are displayed for probe packets on individual VIPs and the total number of probe packets for all VIPs configured for the subinterface.

Example

```
--{ candidate shared default }--[  ]--
# info from state with-context interface irb1 subinterface 1 ipv4 arp
    interface irb1 {
        subinterface 1 {
            ipv4 {
                arp {
                    virtual-ipv4-discovery {
                        address 10.10.10.10 {
                            probe-bridged-subinterfaces [
                                ethernet-1/1.1
                            ]
                            probe-interval 5
                            allowed-macs [
                                00:14:9C:78:E2:E1
                                00:14:9C:78:E2:E2
                                00:14:9C:78:E2:E3
                            ]
                            statistics {
```


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

```
                                out-probe-packets 100
                            }
                        statistics {
                            out-total-probe-packets 100
                        }
                    }
                }
            }
        }
    }
```

4.6.8 Layer 3 proxy-ARP/ND

Note:
The components of EVPN Layer 3 are supported on 7220 IXR-D*x*, 7250 IXR Gen 2, 7250 IXR Gen 2c+, 7250 IXR Gen 3, 7220 IXR-H5, and 7220 IXR-H6 platforms.

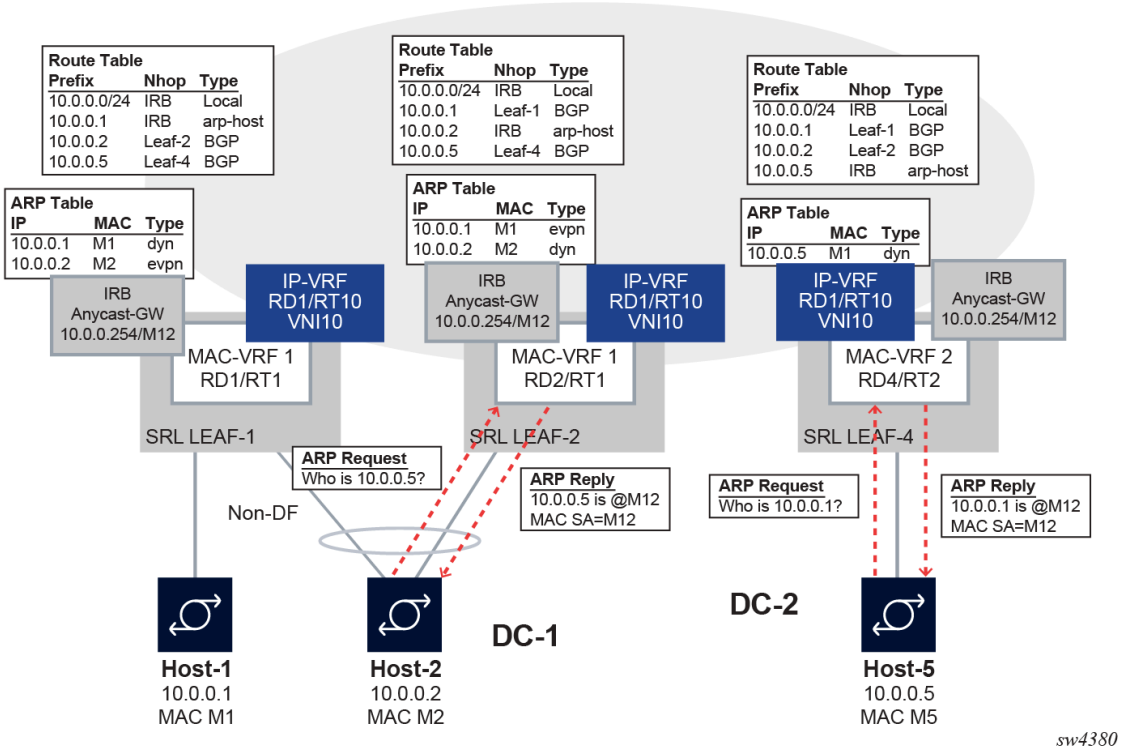
Layer 3 proxy-ARP/ND configures the router to reply with its own MAC to ARP-requests and NS destined for any host. This feature is intended for the following use cases:

- Deployments that use a Layer 3 multihoming solution, but are unable to run EVPN, ESs or VXLAN/MPLS.
- Virtual subnet scenarios, where leaf nodes are attached to different broadcast domains (BDs), but the hosts connected to them are part of the same subnet. RFC 7814 describes this type of solution.

The following figure shows an example virtual subnet configuration that uses this feature.

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

Figure 22: Layer 3 proxy ARP/ND



In this example, SR Linux Leaf-1/Leaf-2 and SR Linux-4 are attached to different BDs; however, the hosts connected to them are part of the same subnet. This type of configuration is typically used in scenarios where the BDs are kept deliberately small, while hosts of the same subnet can be part of more than one of the BDs. This is the case described in RFC 7814.

Layer 3 proxy-ARP/ND triggers the router to reply to any ARP-request or NS from hosts, so that the traffic is attracted to the IRB subinterface and forwarded by the router to a destination on the same subinterface from which the request came or a different subinterface or route.

Layer 3 proxy-ARP/ND is intended to be used along with the `learn-unsolicited` and `host-route populate` features, so that ARP/ND entries are generated and host routes created out of them and advertised by the routing protocol enabled in the routing network instance (default or IP-VRF).

This feature is supported on IRB and other Layer 3 subinterfaces. EVPN may or may not be enabled on the attached MAC-VRF. The subinterfaces can be attached to an IP-VRF or the default network instance.

Note: The virtual subnet expansion enabled by this feature only works for unicast IP traffic with TTL > 1 between hosts in different BDs.

4.6.8.1 Layer 3 proxy-ARP

When Layer 3 proxy-ARP is enabled, all ARP-requests are copied to the CPM and replied using the `anycast-gw` MAC, if configured; otherwise, the interface `hw-mac-address` MAC is used. This process is irrespective of the target IP address of the ARP-request being in the ARP table or in the route-table.

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

The system does not reply to gratuitous ARP (GARP) requests, where the ARP Sender Protocol Address and ARP Target Protocol Address are both set to the IP address of the cache entry to be updated.

When Layer 3 proxy-ARP is enabled, the router replies to any ARP-request unconditionally if received over a bridged subinterface. If the ARP-request is received over VXLAN/MPLS, the router replies only if the ARP-request is targeted to its own IP; if it is targeted to a non-local IP on the IRB, the ARP-request is flooded to local bridged subinterfaces, changing the source MAC and IP with the local ones.

If the original ARP-request came on a bridged subinterface, and the target IP has not been learned on the subinterface yet, the router sends an ARP-request with its own source information to get the target IP learned as soon as possible.

The enabled proxy-ARP behavior attracts all traffic from all the hosts attached to the IRB subinterface that sent an ARP-request for the destination IP address. As a result of this:

- If the received packets have an IP DA with an ARP entry on the same subinterface over which the ARP-request was received, the router forwards the packets to the next-hop indicated by the ARP entry.
- If the received packets have an IP DA with an ARP entry on a different subinterface, the router forwards the packets to the next-hop indicated by the ARP entry.
- If the received packets have an IP DA with a matching route in the route-table, the router forwards the packets based on the route-table entry.
- Otherwise, the attracted traffic is dropped.

The Layer 3 proxy-ARP feature does not require a route lookup before replying to the received ARP-requests. The received ARP-requests are only copied to CPM; they are not flooded to other objects in the MAC-VRF.

The ICMP redirects generated by XDP CPM are suppressed for IP packets targeting an IP address within the same subnet.

Received ARP-replies with a DA not equal to the local MAC, if any are received, are forwarded based on a mac-table lookup.



Note: The system verifies the source IP and replies regardless of the target IP (it floods to bridged subinterfaces if received on VXLAN/MPLS tunnels). The command `learn-unsolicited` is required to create an ARP entry.

4.6.8.1.1 Configuring Layer 3 proxy-ARP

Procedure

To configure Layer 3 proxy-ARP, enable it on a subinterface.

Example

```
--{ candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 subinterface 1
    interface ethernet-1/1 {
        subinterface 1 {
            ipv4 {
                admin-state enable
                arp {
                    proxy-arp true
                }
            }
        }
```


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

```
    }
```

4.6.8.2 Layer 3 proxy-ND

When Layer 3 proxy-ND is enabled, it functions similarly to Layer 3 proxy-ARP. All NS messages sent to the Solicited-Node multicast address `[ff02::1:ff][low-3-bytes-unicast-ip]` are copied to CPM, and all of the NS messages are replied using the `anycast-gw` MAC, if configured (or interface `hw-mac-address` otherwise), without any route lookup.

Received NS messages are only copied to CPM and not flooded in the BD. In this way, the feature provides an implicit ND flood suppression.

The **learn-unsolicited** command is not needed to learn a neighbor from an NS, because it is basic ND behavior to create a neighbor for the SA of an NS if there is a reply and it is a valid neighbor.

4.6.8.2.1 Configuring Layer 3 proxy-ND

Procedure

To configure Layer 3 proxy-ND, enable it on a subinterface.

Example

```
--{ candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 subinterface 1 ipv6 neighbor-discovery
    interface ethernet-1/1 {
        subinterface 1 {
            ipv6 {
                neighbor-discovery {
                    proxy-nd true
                }
            }
        }
    }
```

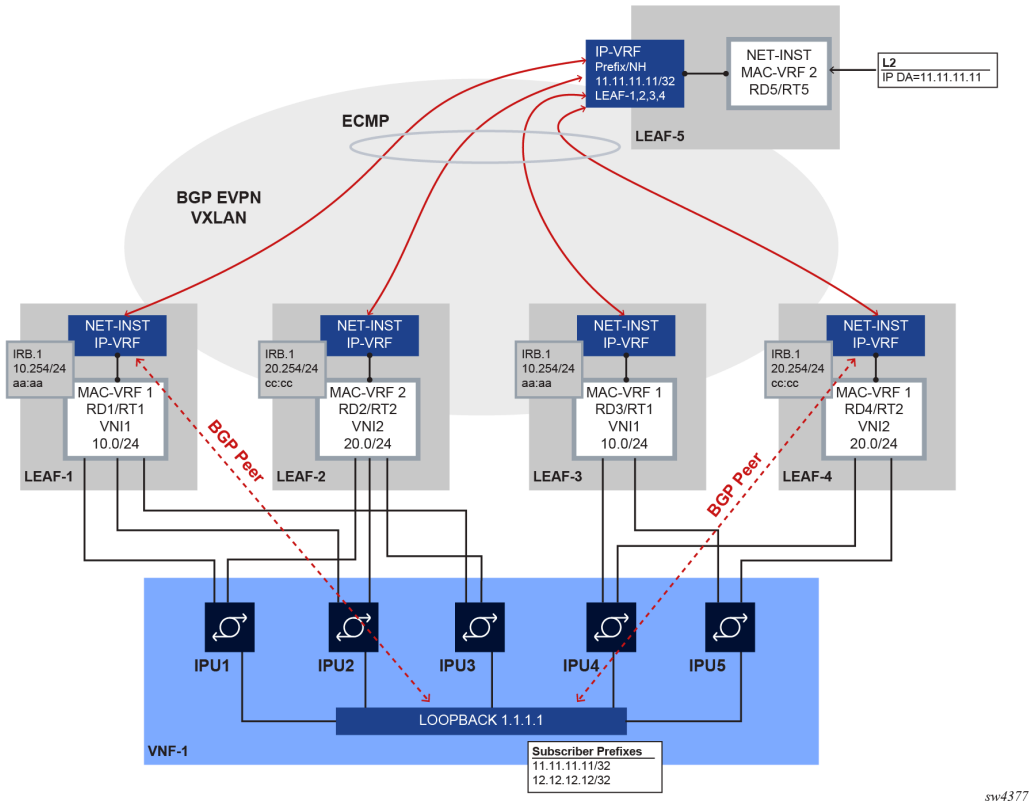
4.6.9 EVPN IP aliasing

As defined in *draft-ietf-bess-evpn-ip-aliasing*, IP aliasing can be used with EVPN Layer 3 Ethernet Segments (ESs) to load-balance traffic to a host connected to multiple leaf nodes attached to the same IP-VRF, even if some of the leaves do not advertise reachability to that host.

SR Linux supports using IP aliasing for BGP peering where leaf switches are multihomed to a single virtual/container network function (referred to as VNF in this section) loopback address. The following figure shows this type of configuration.

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

Figure 23: BGP peering to VNF loopback



Note:
EVPN-VXLAN is used to describe IP aliasing in this section, however IP aliasing is also applicable to EVPN-MPLS services.

In this configuration, leaf nodes LEAF-1 through LEAF-5 are connected to an EVPN Layer 3 tenant domain, and four of these leaf nodes are multihomed to the same VNF. This VNF has five IPUs, similar to line cards, that are each dual-homed to two leaf nodes. Each IPU has two Layer 3 links that are connected to the leaf switches.

MAC-VRF 1 and MAC-VRF 2 are different Broadcast Domains (BDs) and have no Layer 2 EVPN connectivity between them.

The VNF is attached to a pair of subscriber prefixes, 11.11.11.11/32 and 12.12.12.12/32, that must be advertised to the leaf nodes, peering from the IP-VRF IRB interfaces, so the leaf nodes can distribute the reachability to the rest of the network.

In this example, the VNF can run PE-CE BGP from loopback interface 1.1.1.1/32, but supports a maximum of two BGP sessions.

In this configuration, IP aliasing on SR Linux allows the following:

- Full traffic spraying in both upstream and downstream directions, even though the number of BGP sessions is limited to two, and the VNF connects to four leaf nodes.

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

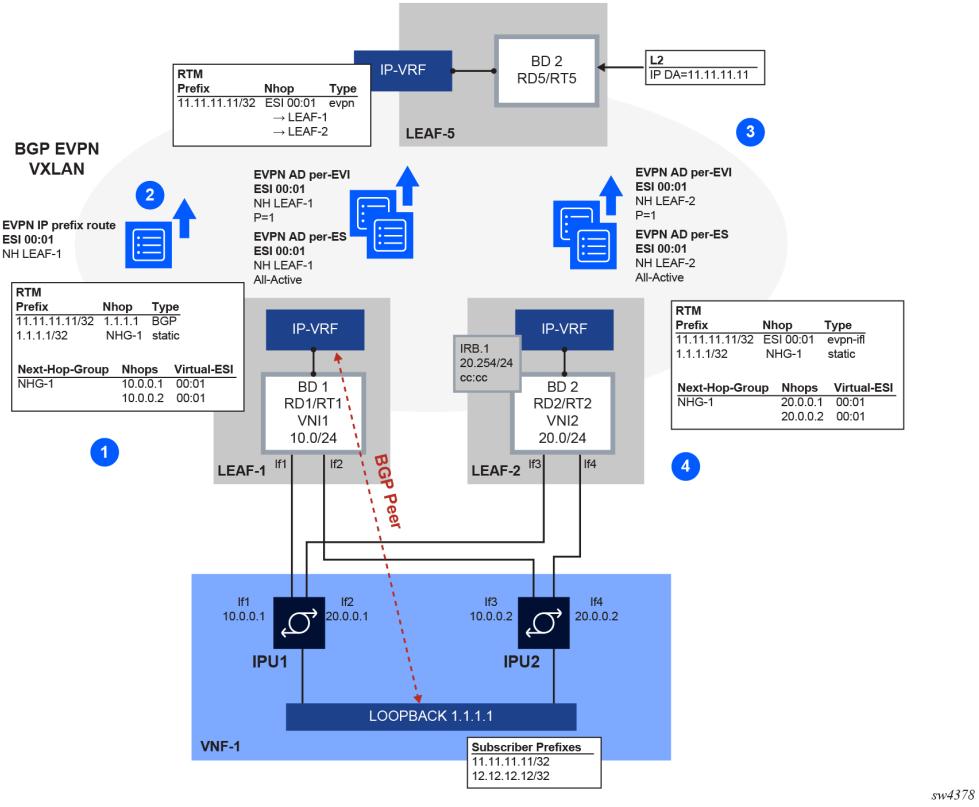
- No tromboning for downstream traffic routed to a leaf node that does not have a BGP session with the VNF. The local leaf must always route locally to the VNF, unless the VNF loopback becomes unreachable.

  For example, when LEAF-5 sends traffic with destination 11.11.11.11 to LEAF-3, LEAF-3 needs to route locally to its local MAC-VRF 1, even if it has an RT5 in its route table pointing at LEAF-1 or LEAF-4.

EVPN IP aliasing on SR Linux

The following figure shows how IP aliasing on SR Linux works for BGP peering to a VNF loopback. To simplify the explanation, only two leaf nodes are shown; however, multiple leaf nodes can be multihomed to the same VNF loopback.

Figure 24: EVPN IP aliasing on SR Linux



In this configuration, BGP peering to the loopback on the VNF works as follows. The numbers represent the steps depicted in the figure.

1. The VNF's loopback address (1.1.1.1) is associated with virtual Layer 3 ES ES-1 on both LEAF-1 and LEAF-2, working in all-active mode. Static routes to 1.1.1.1/32 are configured on both leaf switches. These static routes can run BFD to the VNF, if required, to speed up fault detection.
   After ES-1 is enabled, and as soon as 1.1.1.1 has a route (of any mask length) in the IP-VRF's route table, LEAF-1 and LEAF-2 advertise an EVPN ES route, as well as EVPN Auto Discovery (AD) per ES and per EVI routes for ES-1. The AD routes are advertised based on the presence of a route for 1.1.1.1 in the route table. In this example, the route for 1.1.1.1/32 is statically configured in the route table, but any non-EVPN IP prefix route that provides reachability to 1.1.1.1 would trigger the advertisement of the AD routes.

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

If the ES ES-1 oper-state is up and a route for 1.1.1.1 is active in the route table, each leaf advertises:

- AD per-ES route for ES-1, indicating all-active mode
- AD per-EVI route for ES-1, indicating P = 1 (primary state is set), with route distinguisher and route target of the IP-VRF

The AD per-EVI routes contain two flags, P (primary) and B (backup), which provide an indication for the remote nodes to know whether they can send traffic to the owners of the routes. The remote nodes send traffic for the ES to all the nodes advertising themselves as primary (P = 1), and not to nodes signaling P = 0, B = 1, unless there are no primary nodes left in the ES.

For all-active multihoming mode, there is no DF election in the context of the ES.

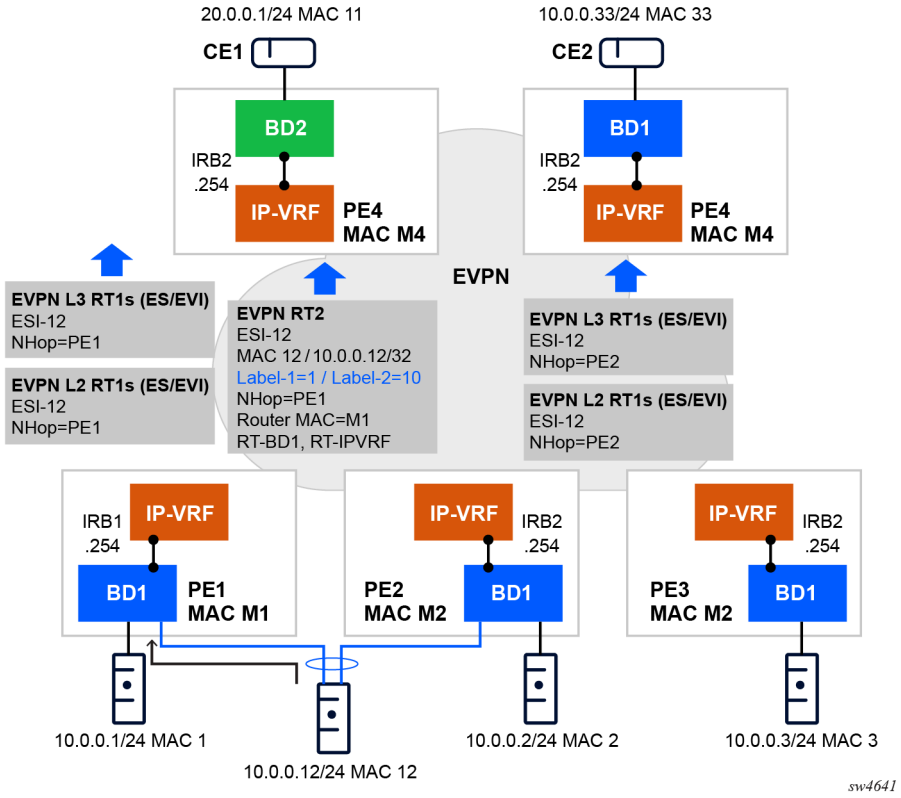
2. Any prefixes that are resolved to the next-hop associated with ESI-1 are advertised as EVPN IP prefix routes with ESI 00:01 (representation of the ESI for ES-1). In the example in Figure 24: EVPN IP aliasing on SR Linux, prefix 11.11.11.11/32 is advertised as an EVPN IP prefix route with ESI 00:01.
3. LEAF-5 identifies the EVPN IP prefix route with ESI 00:01 and installs the route in the IP-VRF route table with as many next-hops as received AD routes with ESI 00:01 and the primary flag set (P = 1). That is, the IP prefix route with ESI 00:01 is recursively resolved to the next hops that advertise the AD per-EVI (with P = 1) and per-ES routes for 00:01.
   If the virtual Layer 3 ES is configured as single-active, a DF Election among all the leaf nodes attached to the same ES determines which leaf nodes advertise themselves as primary for the ES.
4. LEAF-2 identifies the EVPN IP prefix route with ESI 00:01 and next-hop LEAF-1 and resolves the route recursively to the next-hop associated with ES-1.
   If the static route associated with ES-1 goes down, the leaf withdraws the associated Layer 3 AD per-EVI/ES routes to avoid attracting traffic from LEAF-5.

EVPN IP aliasing for BGP-EVPN-IFL host routes

The following figure shows how IP aliasing on SR Linux works for BGP-EVPN-IFL host routes in the symmetric IRB model. In IP aliasing for MAC/IP routes, the existing Layer 2 ES configuration is used.

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

Figure 25: IP aliasing for BGP-EVPN-IFL host routes



In this configuration, the **interface subinterface ipv4 arp/nd evpn advertise dynamic/static interface-less-routing** configuration is used to enable the advertisement of MAC/IP routes as BGP-EVPN-IFL host routes so that they can be imported on the remote nodes IP-VRF network-instances.

The following configuration enables this behavior on an IRB interface.

```
--{ * candidate shared default }--[  ]--
# info with-context interface irb0
    interface irb0 {
        subinterface 1 {
            ipv4 {
                arp {
                    learn-unsolicited true
                    evpn {
                        advertise dynamic {
                            internal-tags {
                                set-tag-set [
                                    set-1
                                ]
                            }
                            interface-less-routing {
                            }
                        }
                    }
                }
            }
        }
```


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

```
    }
```

The configuration of the **advertise-ifl-host-ad-routes** command at the ES level triggers the advertisement of the AD per ES and AD per EVI routes for the ES in the context of the IP-VRF, also adding the RD/RT/label of the IP-VRF. The IP-VRFs are picked from the association to the IRB that is connected to the MAC-VRFs where the ES resides. When the IP-VRF is located, the RD/RT/label/VNI are taken from the instance by the configuration of the **bgp-evpn-instance** command under **advertise-ifl-host-ad-routes**. The default value is **bgp-evpn-instance 1**.

A tag set can be configured under **advertise-ifl-host-ad-routes**. A tag set is used in export policies to match the AD per-EVI routes generated with the Layer 3 RD/RT. If the tag set is not resolved for any reason, for example, `policy-mgr` is down or restarting, the Ethernet Segment goes down.

The following configuration shows the ES with the **advertise-ifl-host-ad-routes** command. This configuration allows remote leaf routers to resolve the EVPN-IFL-HOST routes to the AD per-EVI routes of the same ES.

```
--{ * candidate shared default }--[ system network-instance protocols evpn ethernet-segments
 bgp-instance 1 ethernet-segment ES-1 ]--
A:root@srl1# info with-context
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment ES-1 {
                                admin-state enable
                                esi 01:00:11:11:11:11:00:00:00:00
                                interface ethernet-1/1 {
                                }
                                advertise-ifl-host-ad-routes {
                                    internal-tags {
                                        set-tag-set [
                                            set-2
                                        ]
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

Centralized routing model and PE-CE routes resolved over EVPN-IFL

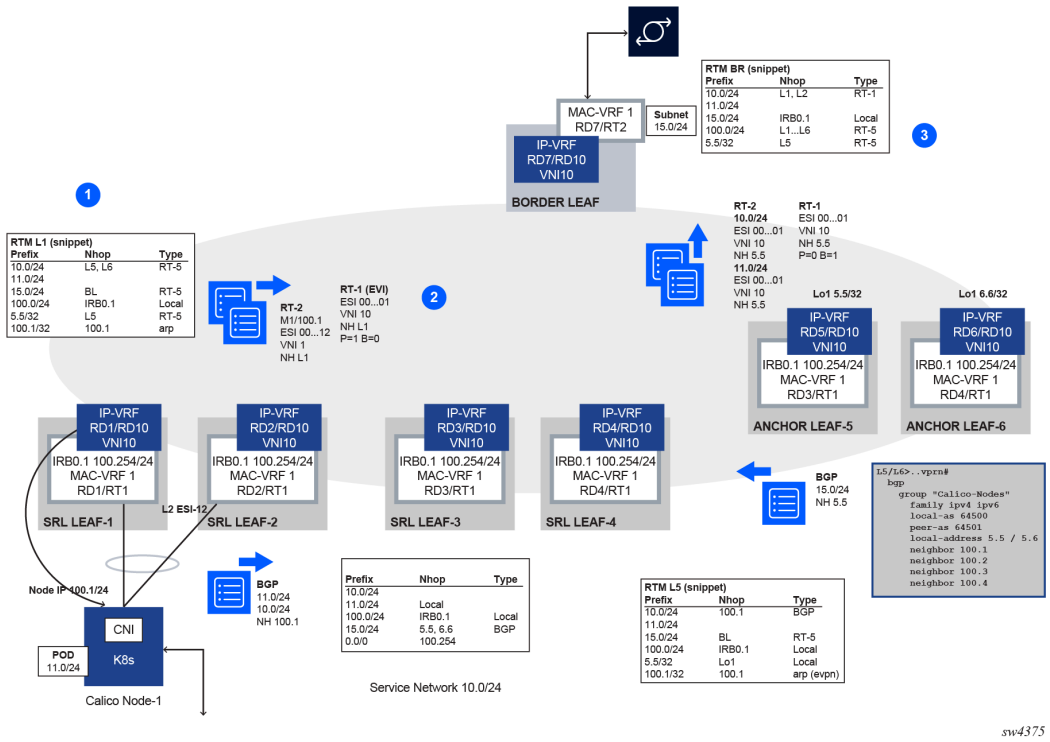
SR Linux supports PE-CE sessions over VXLAN/MPLS and PE-CE next-hop resolution via EVPN-IFL, which, along with Layer 3 ES, enables support for the centralized routing model, shown in Figure 26: Centralized routing model.

This model uses BGP PE-CE peering on CEs to a pair of centralized anchor leaf nodes, as opposed to BGP PE-CE sessions to the directly connected leaf nodes. This is possible because BGP PE-CE route resolution via EVPN IFL routes is allowed, which is necessary when the BGP session is not established via a local subinterface.

Having all the BGP PE-CE sessions centralized in a pair of anchor leaf nodes simplifies the provisioning of BGP, not only on the rest of the leaf nodes, but also on the CEs connected to those leaf nodes.

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

Figure 26: Centralized routing model



In this example, the centralized routing model is used between all the Kubernetes (K8s) node IP addresses and the IP-VRF loopbacks on anchor leaf nodes; for example, between Node IP 100.1/24 (Node-1) and 5.5/32 in ANCHOR LEAF-5. Although only one Kubernetes node is depicted in the figure, multiple nodes may be attached to the leaf nodes throughout the data center. By using this model, the BGP configuration is simplified, especially on the Kubernetes nodes, where they can all use the same configuration with a peer to the anchor leaf nodes.

All the BGP PE-CE routes advertised by the CEs are re-advertised by the anchor leaf nodes using EVPN IP prefix routes. For example, prefixes 10.0/24 and 11.0/24 are advertised by Node-1 and re-advertised by LEAF-5 in IP prefix routes. IP aliasing solves the issue of tromboning the traffic with destination 10.0 or 11.0 from the BORDER LEAF to LEAF-1 or LEAF-2 via anchor LEAF-5. IP aliasing works as follows (where the numbers represent the steps depicted in Figure 26: Centralized routing model):

1. LEAF-1, LEAF-2, LEAF-5 and LEAF-6 are all configured with the virtual Layer 3 ES ES-1, which uses ESI 00..01, is associated with Node-1 IP 100.1, and is configured as single-active. Node-1 advertises BGP PE-CE routes for 10.0/24 and 11.0/24 to LEAF-5.
2. Because LEAF-1 and LEAF-2 have 100.1 in their route table as a local route and they are configured as primary nodes for the ES, they advertise AD per-EVI routes for ES-1 with P = 1. Anchor LEAF-5 and LEAF-6 are configured as non-primary, so even if they have 100.1 in their route table, they advertise their AD per-EVI routes with P = 0 and B = 1 as backup nodes for ES-1. For example, when LEAF-5 receives the BGP PE-CE routes from Node-1, it re-advertises them in EVPN IP prefix routes with ESI 00..01.This is because the BGP PE-CE routes' next-hop matches the IP address in ES-1.
3. BORDER LEAF imports the EVPN IP prefix routes with ESI 00..01 and recursively resolves them to the next-hops of all the AD per-EVI routes advertised as primary (P = 1); that is, LEAF-1 and LEAF-2.

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

When receiving traffic to 10.0/24 or 11.0/24, BORDER LEAF load balances the traffic to only LEAF-1 and LEAF-2, as opposed to the anchor leaf nodes.

Static route trigger for non-zero ESI IP prefix routes

Static route trigger for non-zero ESI IP prefix routes modifies IP aliasing so that the node advertising the IP prefix route can be based on both BGP PE-CE routes and static routes. This modification is present on the transmission side, while the resolution process does not change. No CLI knob is needed to enable this feature.

BGP reacts only to the best route. For example, if the ES next-hop is 1.1.1.1 and there is a BGP route, an OSPF route, and a static route to 10.0.0.0/24 with the next-hop 1.1.1.1, based on the route table preference, the static route is picked and the RT5 is triggered based on that static route.

Static route trigger for non-zero gateway-IP routes also modifies gateway-IP load balancing. For more information, see Gateway IP-based load balancing for EVPN IFL routes.

4.6.9.1 Configuring Layer 3 Ethernet Segments for EVPN IFL IP aliasing

To configure a Layer 3 Ethernet Segment for IP aliasing, you specify the Layer 3 next hop to be associated with an Ethernet Segment (ES). The AD per-ES/EVI routes for the ES are advertised when the specified next-hop address is active in the route table of the IP-VRF with the configured EVI.

The following example configures two examples of Layer 3 Ethernet Segments, which are associated with an IPv4 and an IPv6 next-hop, respectively.

```
A:Leaf-1# info from running system network-instance protocols evpn ethernet-segments
  system {
    network-instance {
      protocols {
        evpn {
          ethernet-segments {
            timers {
              boot-timer 180
              activation-timer 3
            }
            bgp-instance 1 {
              ethernet-segment es_v4_1 {
                type virtual
                admin-state enable
                esi 00:bc:de:00:00:00:00:01:00:04
                multi-homing-mode single-active
                next-hop 6.0.1.254 {
                  evi 1 {
                  }
                }
                df-election {
                  algorithm {
                    type preference
                    preference-alg {
                      preference-value 2
                      capabilities {
                        ac-df exclude
                        non-revertive false
                      }
                    }
                  }
                }
                routes {
```


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

```
                next-hop use-system-ipv4-address
                ethernet-segment {
                  originating-ip use-system-ipv4-address
                }
              }
            }
            ethernet-segment es_v6_1 {
              type virtual
              admin-state enable
              esi 00:bc:de:00:00:00:00:01:00:06
              multi-homing-mode single-active
              next-hop 6:0:1::254 {
                evi 1 {
                }
              }
              df-election {
                algorithm {
                  type preference
                  preference-alg {
                    preference-value 2
                    capabilities {
                      ac-df exclude
                      non-revertive false
                    }
                  }
                }
              }
              routes {
                next-hop use-system-ipv4-address
                ethernet-segment {
                  originating-ip use-system-ipv4-address
                }
              }
            }
          }
        }
      }
    }
  }
```

Layer 3 ESs must be configured as type `virtual` and must be associated with a next-hop and EVI. When the next-hop IP address is active in an IP-VRF identified by the configured EVI, the corresponding AD-per-ES and per-EVI routes for the ES are advertised.

If the Layer 3 ES is configured as `multi-homing-mode all-active`, the DF election algorithm configuration is irrelevant, because there is no DF and all the nodes are primary (they all advertise P = 1 in their AD per-EVI routes). However, if configured as `multi-homing-mode single-active`, a DF election is computed among all the nodes advertising the same ESI in the ES route. In this case, the `default`, `preference`, or `manual` algorithms can be used. The manual DF election simply controls whether the node is primary or not by configuration.

For example, Leaf-1 is configured as primary, so it advertises an AD per-EVI route with P = 1 irrespective of the other leaf nodes in the ES:

```
--{ +* candidate shared default }--[ system network-instance protocols evpn ethernet-segments
 bgp-instance 1 ethernet-segment L3-ES-1 ]--
A:Leaf-1# info
  type virtual
  admin-state enable
  esi 01:20:20:20:01:00:00:00:00:00
  multi-homing-mode single-active
```


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

```
  next-hop 20.20.20.1 {
    evi 2 {
    }
  }
  df-election {
    algorithm {
      type manual
      manual-alg {
        primary-evi-range 2 {
          end-evi 2
        }
      }
    }
  }
```

However, Leaf-2 in the following example is not configured as primary, so it advertises an AD per-EVI route with P = 0 and B = 1. Therefore, it does not attract traffic to the ES as long as there are primary nodes in the ES.

```
--{ +* candidate shared default }--[ system network-instance protocols evpn ethernet-segments
 bgp-instance 1 ethernet-segment L3-ES-1 ]--
A:Leaf-2# info
  type virtual
  admin-state enable
  esi 01:20:20:20:01:00:00:00:00:00
  multi-homing-mode single-active
  next-hop 20.20.20.1 {
    evi 2 {
    }
  }
  df-election {
    algorithm {
      type manual
      manual-alg {
      }
    }
  }
```

The following example shows the configuration of LEAF-1 and LEAF-2 in the Figure 23: BGP peering to VNF loopback figure. LEAF-5 is configured without any ES and with an IP-VRF that has BGP-EVPN enabled and **ecmp** greater than or equal to 2, so that it can load-balance traffic to the two leaf nodes attached to the ES.

```
# LEAF-1 and LEAF-2 configuration of the Ethernet Segment
--{ + candidate shared default }--[  ]--
A:leaf1/leaf2# info with-context system network-instance protocols evpn
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment L3-ES-1 {
                                type virtual
                                admin-state enable
                                esi 01:01:00:00:00:00:00:00:00:00
                                multi-homing-mode all-active
                                next-hop 1.1.1.1 {
                                    evi 2 {
                                    }
                                }
```


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

```
                            }
                        }
                    }
                }
            }
        }
    }

# LEAF-1 IP-VRF configuration, including BGP session to the VNF
--{ + candidate shared default }--[  ]--
A:leaf1# info with-context network-instance ip-vrf-2
network-instance ip-vrf-2 {
  type ip-vrf
  admin-state enable
  interface irb0.1 {
  }
  interface lo1.1 {
  }
  vxlan-interface vxlan1.2 {
  }
  protocols {
    bgp-evpn {
      bgp-instance 1 {
        admin-state enable
        vxlan-interface vxlan1.2
        evi 2
        ecmp 4
      }
    }
    bgp {
        autonomous-system 64501
        router-id 10.0.0.1
        group PE-CE {
        export-policy all
        import-policy all
        peer-as 64500
      }
        afi-safi ipv4-unicast {
            admin-state enable
        }
      neighbor 1.1.1.1 {
        peer-group PE-CE
        multihop {
          admin-state enable
          maximum-hops 10
        }
        transport {
          local-address 10.0.0.1
        }
      }
    }
    bgp-vpn {
      bgp-instance 1 {
        route-distinguisher {
          rd 1.1.1.1:2
        }
        route-target {
          export-rt target:64500:2
          import-rt target:64500:2
        }
      }
    }
  static-routes {
    route 1.1.1.1/32 {
```


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

```
      next-hop-group NHG-1
    }
  }
  next-hop-groups {
    group NHG-1 {
      nexthop 1 {
        ip-address 10.0.0.1
      }
      nexthop 2 {
        ip-address 10.0.0.2
      }
    }
  }
}

# LEAF-2 IP-VRF configuration
--{ + candidate shared default }--[  ]--
A:leaf2# info with-context network-instance ip-vrf-2
network-instance ip-vrf-2 {
  type ip-vrf
  interface irb0.1 {
  }
  vxlan-interface vxlan1.2 {
  }
  protocols {
    bgp-evpn {
      bgp-instance 1 {
        vxlan-interface vxlan1.2
        evi 2
        ecmp 4
      }
    }
    bgp-vpn {
      bgp-instance 1 {
        route-distinguisher {
          rd 2.2.2.2:2
        }
        route-target {
          export-rt target:64500:2
          import-rt target:64500:2
        }
      }
    }
  static-routes {
    route 1.1.1.1/32 {
      next-hop-group NHG-1
    }
  }
  next-hop-groups {
    group NHG-1 {
      nexthop 1 {
        ip-address 20.0.0.1
      }
      nexthop 2 {
        ip-address 20.0.0.1
      }
    }
  }
}
```


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

The following shows the route table for one of the subscriber prefixes (11.11.11.11/32) that are advertised from the VNF with a next-hop matching the Layer 3 ES next-hop. For LEAF-1, the route is learned via BGP PE-CE.

```
# LEAF-1 route table entry
--{ + candidate shared default }--[ network-instance ip-vrf-2 ]--
A:leaf1# info from state with-context route-table ipv4-unicast route 11.11.11.11/32 id 0 route-
type bgp route-owner bgp_mgr
  route-table {
    ipv4-unicast {
      route 11.11.11.11/32 id 0 route-type bgp route-owner bgp_mgr {
        metric 0
        preference 170
        active true
        last-app-update 2022-10-19T10:26:51.020Z
        next-hop-group 423577430891
        next-hop-group-network-instance ip-vrf-2
        resilient-hash false
        fib-programming {
          last-successful-operation-type add
          last-successful-operation-timestamp 2022-10-19T10:26:51.021Z
          pending-operation-type none
          last-failed-operation-type none
        }
      }
    }
  }
```

For LEAF-2, the route is learned via EVPN; however, the next-hop is resolved locally to the next-hop-group associated with the route for 1.1.1.1 (the ES next-hop IP address).

```
# LEAF-2 route table entry
--{ +* candidate shared default }--[ network-instance ip-vrf-2 ]--
A:leaf2# info from state with-context route-table ipv4-unicast route 11.11.11.11/32 id 0 route-
type bgp-evpn route-owner bgp_evpn_mgr
  route-table {
    ipv4-unicast {
      route 11.11.11.11/32 id 0 route-type bgp-evpn route-owner bgp_evpn_mgr {
        metric 0
        preference 170
        active true
        last-app-update 2022-10-19T10:25:06.199Z
        next-hop-group 423577389148
        next-hop-group-network-instance ip-vrf-2
        resilient-hash false
        fib-programming {
          last-successful-operation-type add
          last-successful-operation-timestamp 2022-10-19T10:25:06.199Z
          pending-operation-type none
          last-failed-operation-type none
        }
      }
    }
  }

--{ +* candidate shared default }--[ network-instance ip-vrf-2 ]--
A:leaf2# info from state with-context route-table next-hop-group 423577389148
  route-table {
    next-hop-group 423577389148 {
      backup-next-hop-group 0
      next-hop 0 {
        next-hop 423577389136
```


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

```
        resolved true
      }
      next-hop 1 {
        next-hop 423577389137
        resolved true
      }
    }
  }

--{ +* candidate shared default }--[ network-instance ip-vrf-2 ]--
A:leaf2# show route-table next-hop 423577389136
-------------------------------------------------------------------------------
Next-hop route table of network instance ip-vrf-2
-------------------------------------------------------------------------------
Index : 423577389136
Next-hop : 20.20.20.1
Type : indirect
Subinterface : N/A
Resolving Route : 20.20.20.0/24 (local)
```

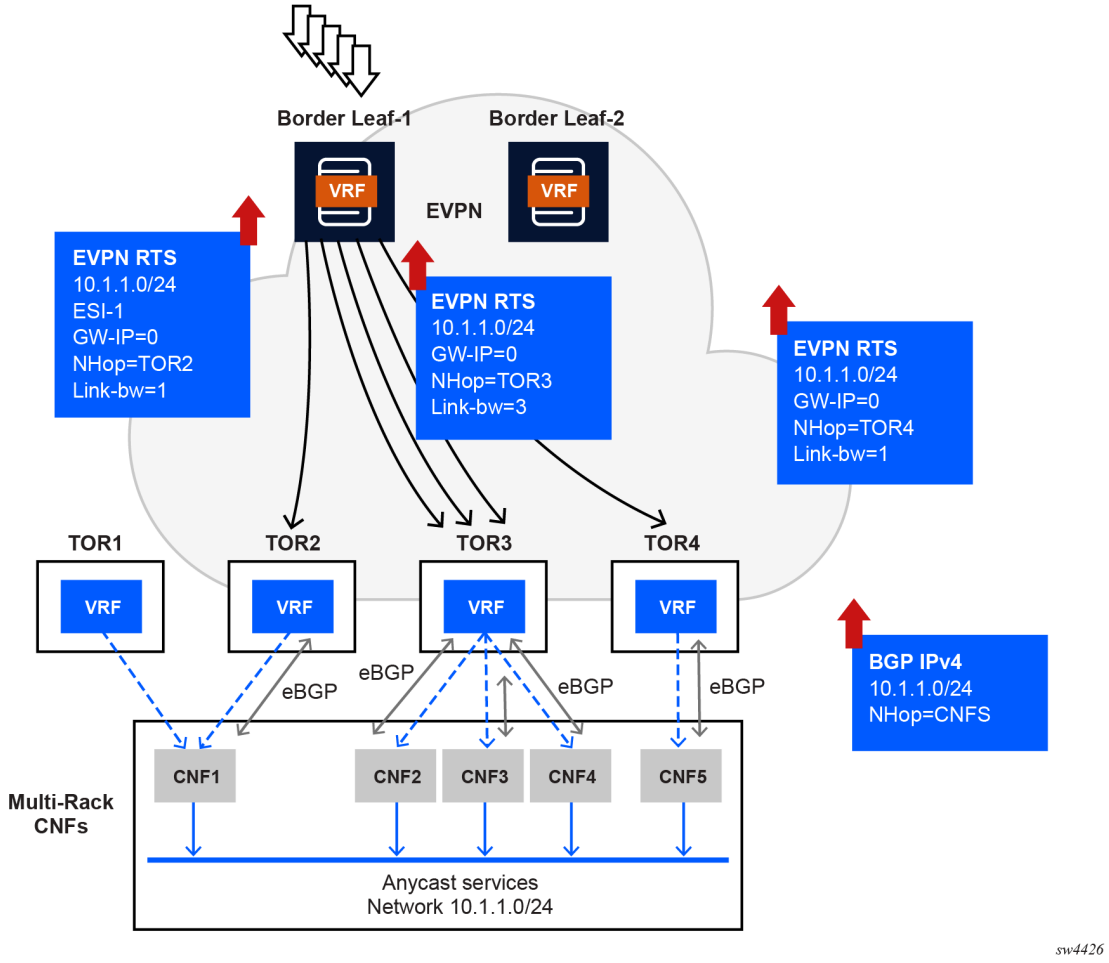
4.6.10 Unequal ECMP for EVPN IP prefix routes

SR Linux supports weighted Equal Cost Multi Path (ECMP) for EVPN IP prefix IFL (interface-less) routes for EVPN-VXLAN and EVPN-MPLS services. To do this, SR Linux makes use of the EVPN link-bandwidth extended community (EC) defined in draft-ietf-bess-evpn-unequal-lb. This extended community indicates the weight for a specific IP prefix; that is, the number of PE-CE multi-paths for an IP prefix that is re-advertised into an EVPN IP prefix route.

Figure 27: EVPN using weighted ECMP shows an example of weighted ECMP. Assuming each Container Network Function (CNF) advertises the anycast subnet 10.1.1.0/24 from a different next-hop, each TOR ends up with a different number of multi-paths in its PE-CE session. In the example below, TOR3 has three multi-paths for the anycast subnet, and the advertised EVPN IP prefix route includes an EVPN link-bandwidth extended community with a weight of 3. The rest of the TORs send a weight of 1. Note that TOR1 and TOR2 are multi-homed to the same CNF1, yet they send a weight of 1.

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

Figure 27: EVPN using weighted ECMP



On the border leafs / data center gateways, when this feature is enabled, if the EVPN IP prefix route has an Ethernet Segment Identifier (ESI) of 0, the PE sprays the flows to the EVPN IP prefix route based on the received weight; in this example, one-fifth of the flows are sent to TOR4, and three-fifths are sent to TOR3.

When the EVPN IP prefix route has a non-zero ESI, and there is a weight in the route:

- The EVPN link-bandwidth extended community received in the EVPN IP prefix route indicates the weight for the EVPN IP prefix route.
- The PE sprays the flows to the EVPN IP prefix route based on the received weight, dividing the flows to an ES among the number of PEs attached to the ES.

  In the example above, one-fifth of the flows are sent to the aliased pair TOR1/TOR2 (either one is selected because TOR1 and TOR2 are attached to the same Ethernet Segment).
- The system rounds up when the advertised weight for the IP Prefix is a non-zero ESI, divided by the number of PEs in the ES, is not an integer.

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

For example, if ES1 (TOR1/TOR2) advertises BW=3 on the IP Prefix route, and TOR4 advertises BW=1, then 3 (BW) / 2 (PE in ES1) = 1.5. The system rounds up, and the remote nodes install weight=2 for TOR1, weight=2 for TOR2, and weight=1 for TOR4.

- If the weight received in a non-zero ESI IP prefix route exceeds 128, the system caps it at 128, then divides the weight into the number of PEs in the ES.
- If two EVPN IP prefix routes are received for the same prefix, same ESI, different route distinguishers (RDs), they should have the same weight. However, if they have different weights, the system selects the weight from the first EVPN IP prefix route.
- If the EVPN link-bandwidth extended community is missing from any of the PEs in an ECMP set, or the Value Units field of the extended community is inconsistent, the weight is ignored by the receiving PE, and regular ECMP forwarding is performed. The Value Units field can indicate "bandwidth" or a "generalized weight", with only the latter being supported by SR Linux.

4.6.10.1 Advertising the EVPN link-bandwidth extended community

Procedure

To configure weighted ECMP, you enable the advertisement of the EVPN link-bandwidth extended community and specify the weight to be advertised in the extended community.

You can configure the following parameters for the advertised weight in the extended community:

- `weight` specifies the weight to be advertised in the EVPN link-bandwidth extended community for the advertised EVPN IP prefix routes for the service. If set to `dynamic` (the default value), the weight is dynamically set based on the number of BGP PE-CE paths for the IP prefix that is advertised in an EVPN IP prefix route. The dynamic weight only considers BGP PE-CE paths.

  Alternatively, the weight can be set to a fixed integer value in the range 1 to 128.
- `maximum-dynamic-weight` specifies the maximum weight to be advertised in the EVPN link-bandwidth extended community for the advertised EVPN IP-Prefix routes for the service. If weight `dynamic` is configured, the actual advertised weight is the minimum of the number of BGP PE-CE paths for the prefix and the configured `maximum-dynamic-weight`.

Example

The following example enables advertisement of the EVPN link bandwidth extended community and specifies the weight to be included in the extended community for the advertised EVPN IP prefix routes.

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance VRF1 protocols bgp-evpn bgp-instance 1
    network-instance VRF1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        route-table {
                            ip-prefix {
                                evpn-link-bandwidth {
                                    advertise {
                                        weight 60
                                    }
                                }
                            }
```


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

```
                        }
                    }
                }
            }
        }
    }
```

4.6.10.2 Enabling weighted ECMP

Procedure

When weighted ECMP is enabled, the system takes into account the EVPN link-bandwidth extended community when installing an ECMP set for an EVPN IP prefix route in the IP-VRF route table.

Flows to an IP prefix received with a weight and a zero ESI are sprayed according to the weight. If the EVPN IP prefix route received with a weight has a non-zero ESI, the weight is divided into the number of PEs attached to the ES, and rounded up if the result is not an integer.

The command also enables weighted ECMP for BGP CEs that are configured with a weight specified with the `link-bandwidth.add-next-hop-count-to-received-bgp-routes` setting.

Configuring `max-ecmp-hash-buckets-per-next-hop-group` preserves the datapath resources used for the weighted next-hops. The normalization algorithm also refers to this number of hash buckets. See Displaying ECMP weights for an example of how the weights are normalized the using the `max-ecmp-hash-buckets-per-next-hop-group` setting.

Example

The following example enables weighted ECMP and specifies the maximum number of ECMP hash buckets per next-hop-group. The weights for weighted ECMP are normalized based on this number of hash buckets.

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance VRF1 protocols bgp-evpn bgp-instance 1
    network-instance VRF1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        route-table {
                            ip-prefix {
                                evpn-link-bandwidth {
                                    weighted-ecmp {
                                        admin-state enable
                                        max-ecmp-hash-buckets-per-next-hop-group 4
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```


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

Example

Note: The range of the command `max-ecmp-hash-buckets-per-next-hop-group` differs depending on the platform. Validation of the configuration prevents the configuration of a value that the platform cannot support.

4.6.10.3 Configuring weighted ECMP for received PE-CE BGP routes

Procedure

When the system is enabled to process the link-bandwidth extended community, you can configure a weight to be internally added to the received PE-CE BGP routes for the purpose of EVPN unequal ECMP.

Example

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance VRF1 protocols bgp group g1 link-bandwidth
    network-instance VRF1 {
        protocols {
            bgp {
                group g1 {
                    link-bandwidth {
                        add-next-hop-count-to-received-bgp-routes 100
                    }
                }
            }
        }
    }
```

4.6.10.4 Displaying ECMP weights

Procedure

The system normalizes the weights used in weighted ECMP using the `max-ecmp-hash-buckets-per-next-hop-group` setting and the advertised weights, according to the algorithm described in "Normalizing datapath weights" in the *SR Linux Routing Protocols Guide*.

You can display the weights for a next-hop group.

Example

In the following example, the same prefix 19.1.1.1/32 is received from two PEs with weights 40 and 60, respectively. The maximum supported ECMP paths is 8. The `max-ecmp-hash-buckets-per-next-hop-group` setting is 4. The system programs the next-hops with normalized weights.

To display the number of the next-hop-group for prefix 19.1.1.1/32:

```
--{ running }--[  ]--
# info from state with-context network-instance VRF1 route-table ipv4-unicast route
 19.1.1.1/32 id 0 route-type bgp-evpn route-owner bgp_evpn_mgr origin-network-instance
 VRF1
    network-instance VRF1 {
        route-table {
            ipv4-unicast {
                route 19.1.1.1/32 id 0 route-type bgp-evpn route-owner bgp_evpn_mgr
 origin-network-instance VRF1 {
                    leakable false
```


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

```
                    metric 0
                    preference 170
                    active true
                    last-app-update 2023-05-31T18:29:25.207Z
                    next-hop-group 94413408183
                    next-hop-group-network-instance VRF1
                    resilient-hash false
                    fib-programming {
                        suppressed false
                        last-successful-operation-type modify
                        last-successful-operation-timestamp 2023-05-31T18:29:25.207Z
                        pending-operation-type none
                        last-failed-operation-type none
                    }
                }
            }
        }
    }
```

To display the received weights for the individual next hops in the next-hop group:

```
--{ running }--[  ]--
# info from state with-context network-instance VRF1 route-table next-hop-group
 94413408183
    network-instance VRF1 {
        route-table {
            next-hop-group 94413408183 {
                backup-next-hop-group 0
                fib-programming {
                    last-successful-operation-type add
                    last-successful-operation-timestamp 2023-05-31T18:29:25.207Z
                    pending-operation-type none
                    last-failed-operation-type none
                }
                next-hop 0 {
                    next-hop 94413408157
                    weight 40
                    resolved true
                }
                next-hop 1 {
                    next-hop 94413408152
                    weight 60
                    resolved true
                }
            }
        }
    }
```

The system normalizes the weights using the algorithm described in "Normalizing datapath weights" in the *SR Linux Routing Protocols Guide*, taking into account the received weights, the maximum supported ECMP paths, and the `max-ecmp-hash-buckets-per-next-hop-group` setting. However, the normalized weights that are programmed in the data path cannot be displayed.

4.6.11 Resilient hashing for EVPN IFL

Resilient hashing is supported for EVPN IFL where an IFL route points to a next-hop group (NHG) that contains indirect next hops. The indirect next hops are then resolved by MPLS tunnels.

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

The resilient hashing feature has "sticky" flows even when flows are rehashed because of newly added or failed links. This feature prevents rerouting of flows that are unaffected by network link failures.

SR Linux supports resilient hashing for the following routes:

- regular ECMP with EVPN IFL routes
- unequal ECMP with EVPN IFL routes
- combined EVPN IFL and BGP PE-CE weighted ECMP

Resilient hashing is also supported for the addition or removal of next hops.

4.6.11.1 Resilient hashing and unequal ECMP for EVPN IFL

SR Linux supports combined resilient hashing and unequal ECMP for EVPN IP prefix IFL routes. Both resilient hashing and unequal ECMP can be used on the same prefix.

Configuring resilient hashing and unequal ECMP for EVPN IFL

The following is an example configuration for resilient hashing and unequal ECMP for EVPN IFL:

```
--{ * candidate shared default }--[ network-instance ip-vrf-2 ]--
A:leaf1# info with-context network-instance ip-vrf-2
    network-instance ip-vrf-3 {
        type ip-vrf
        admin-state enable
        ip-load-balancing {
            resilient-hash-prefix 10.10.10.0/24 { // resilient hashing is configured for prefix
 10.10.10.0/24
                hash-buckets-per-path 16
                max-paths 16
            }
        }
        interface irb0.1 {
        }
        interface lo1.1 {
        }
        vxlan-interface vxlan1.2 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.2
                    evi 2
                    ecmp 4
                    routes {
                        route-table {
                            ip-prefix {
                                evpn-link-bandwidth {
                                    advertise {
                                        weight dynamic
                                    }
                                    weighted-ecmp { // unequal ECMP is configured for EVPN IFL
                                        admin-state enable
                                        max-ecmp-hash-buckets-per-next-hop-group 128
                                    }
                                }
                            }
                        }
```


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

```
                }
            }
        }
        bgp {
            autonomous-system 64501
            router-id 1.1.1.1
            afi-safi ipv4-unicast {
                admin-state enable
                multipath {
                    max-paths-level-1 4
                    max-paths-level-2 4
                }
            }
            group PE-CE {
                export-policy all
                import-policy all
                peer-as 64500
                afi-safi ipv4-unicast {
                    ipv4-unicast {
                        link-bandwidth {
                            add-next-hop-count-to-received-bgp-routes 1
                        }
                    }
                }
                timers {
                    connect-retry 1
                    hold-time 9
                    keepalive-interval 3
                    minimum-advertisement-interval 1
                }
            }
            neighbor 20.20.20.1 {
                peer-group PE-CE
                multihop {
                    admin-state enable
                    maximum-hops 10
                }
                transport {
                    local-address 1.1.1.1
                }
            }
        }
        bgp-vpn {
            bgp-instance 1 {
                route-distinguisher {
                    rd 1.1.1.1:2
                }
                route-target {
                    export-rt target:64500:2
                    import-rt target:64500:2
                }
            }
        }
    }
```

Resilient hashing is configured for the 10.10.10.0/24 prefix using the **resilient-hash-prefix** parameter. The **hash-buckets-per-path** parameter indicates the number of times each next hop is repeated in the fill pattern. The **max-paths** parameter indicates the maximum number of ECMP next hops per route associated with the **resilient-hash-prefix**. The product of the **hash-buckets-per-path** and **max-paths** parameters is the total number of buckets that are used when applying the normalized weights.

Unequal ECMP is configured for the same EVPN IFL prefix using the **weighted-ecmp** and **max-ecmp-hash-buckets-per-next-hop-group** parameters.

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

To configure both resilient hashing and unequal ECMP on the same prefix, the **hash-buckets-per-path** and **max-ecmp-hash-buckets-per-next-hop-group** parameters can be set to different values. The **hash-buckets-per-path** parameter will override **max-ecmp-hash-buckets-per-next-hop-group**.

Example

The resilient hashing configuration contains the following parameter values:

- **max-paths**=16
- **hash-buckets-per-path**=16

In this example, the unequal ECMP configuration contains the following parameter value: **max-ecmp-hash-buckets-per-next-hop-group**=128

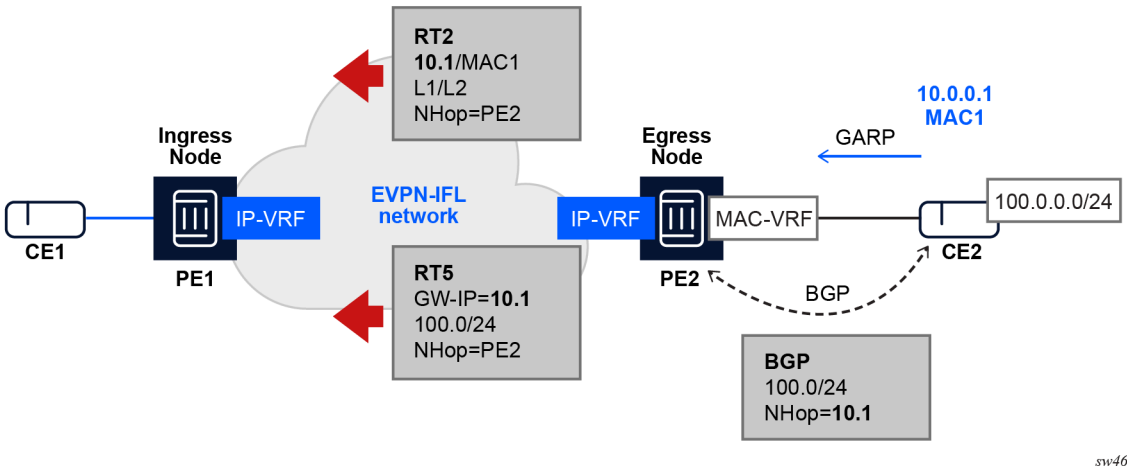
The above values result in the router using 256 hash buckets for the prefix, with an ECMP member cap of 16.

4.6.12 Gateway IP-based load balancing for EVPN IFL routes

Note: This feature is supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, 7250 IXR Gen 3, and 7220 IXR-D*x* platforms.

SR Linux supports recursive resolution of non-zero gateway IP (GW-IP) prefix routes, as described in Section 9.2 of RFC 9135. The GW-IP field in an EVPN IP prefix route carries an IP address that acts as an index for route resolution. In this model, the IP prefix route is resolved using the GW-IP value, instead of through the traditional NLRI next-hop resolution. The following figure shows an example of the recursive resolution of non-zero GW-IP prefix routes.

Figure 28: Recursive resolution of non-zero GW-IP prefix routes



When both the ingress and egress nodes are configured for GW-IP advertisement and resolution, a PE-CE route for prefix 100.0/24 (short for 100.0.0.0/24) is re-advertised by the egress node as an EVPN IP prefix route. In this re-advertised route, the next hop from the original BGP PE-CE route is encoded in the GW-IP field of the IP prefix route.

The ingress node that imports the non-zero GW-IP IP prefix route resolves it to an EVPN-IFL-HOST route (MAC/IP Advertisement route imported in the IP-VRF) whose IP address matches the GW-IP.

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

The advantage of this recursive resolution, or indirection, is that if CE2 moves to a different location and its EVPN-IFL-HOST route is re-advertised from there, the ingress node automatically updates the prefix resolution to reflect the location of the new EVPN-IFL-HOST route. The same prefix can be resolved to multiple GW-IPs when multiple IP prefix routes for the same prefix are received with different GW-IPs. In that case, the ingress node installs the prefix associated with a next hop group that can contain multiple weighted next hops.

The advertisement and resolution of non-zero GW-IP IP prefix routes is configured under the **evpn-interface-less-gateway-ip** container, as follows:

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance ip-vrf
    network-instance ip-vrf {
        type ip-vrf
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        route-table {
                            ip-prefix {
                                evpn-interface-less-gateway-ip {
                                    advertise {
                                    }
                                    resolve {
                                        admin-state enable
                                        max-ecmp-hash-buckets-per-next-hop-group 1
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

The **evpn-interface-less-gateway-ip** command is only available in IP-VRF network instances.

Configuring **advertise** enables the advertisement of non-zero GW-IP addresses in EVPN-IFL routes advertised for the IP-VRF. The GW-IP of the EVPN-IFL routes is populated with the IPv4 or IPv6 next hops of the BGP PE-CE routes programmed in the same IP-VRF.

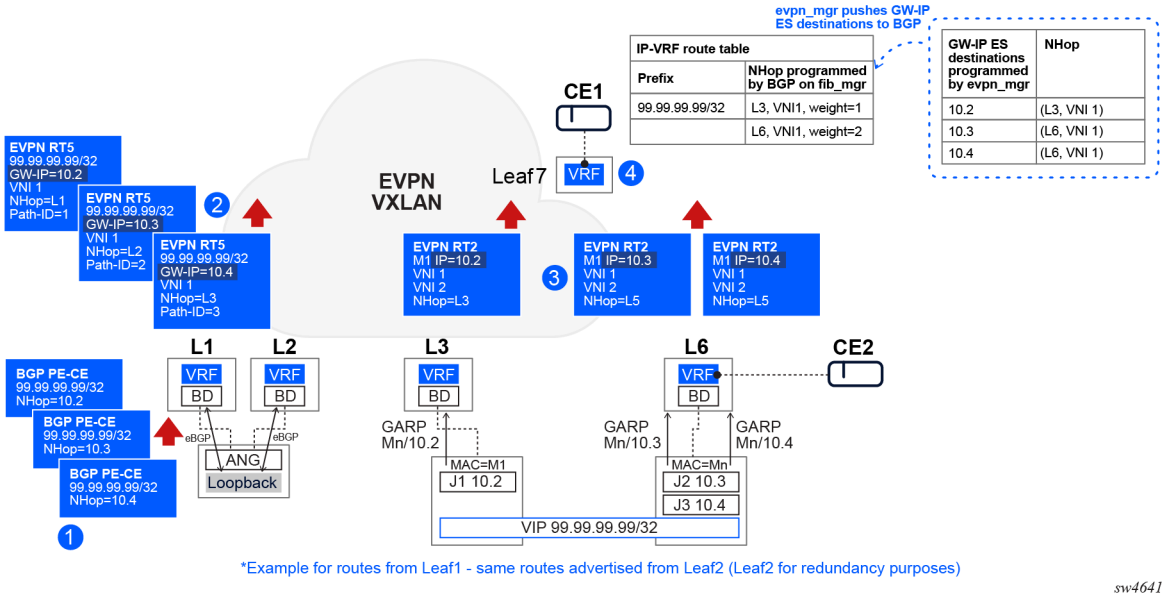
The **resolve** command is disabled by default so that the GW-IP in EVPN-IFL routes is not processed. The **max-ecmp-hash-buckets-per-next-hop-group** command specifies the maximum number of ECMP hash buckets per next hop group created for a specific EVPN-IFL route out of all the EVPN-IFL host routes that resolve the GW-IPs. When **resolve** is enabled, the router resolves IP prefix routes with non-zero GW-IP to an EVPN-IFL-HOST route with an IP that matches the GW-IP or to a local route (IRB interface prefix or ARP-ND route) with an IP that matches the GW-IP.

The **resolve** command can only be enabled when **evpn-link-bandwidth weighted-ecmp** is disabled.

The following figure shows the behavior of GW-IP-based load balancing. In this use case, the non-zero GW-IP IP prefix routes and resolving EVPN-IFL-HOST routes are advertised from different EVPN nodes.

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

Figure 29: Load balancing based on GW-IP



Advertisement of EVPN-IFL routes with non-zero GW-IP

Referring to steps 1 and 2 in Figure 29: Load balancing based on GW-IP, the following occur when **evpn-interface-less-gateway-ip advertise** is configured in nodes L1 and L2:

- When BGP multipath is configured and multiple BGP PE-CE routes are programmed for the same prefix, the router (L1) advertises an EVPN-IFL route per BGP PE-CE route received for the same prefix. In Figure 29: Load balancing based on GW-IP, L1 receives three routes for 99.99.99.99/32 with next hops 10.2, 10.3 and 10.4 respectively, and the three routes are programmed. The **advertise** command triggers the advertisement of three EVPN-IFL routes, as opposed to only one (if **advertise** is not configured).
- For each PE-CE route, an EVPN-IFL route is advertised and encodes the PE-CE route's next hop in the GW-IP field of the EVPN-IFL route. Step 2 in Figure 29: Load balancing based on GW-IP shows the three advertised EVPN-IFL routes with different GW-IPs.
- When multiple EVPN-IFL routes are advertised with the same prefix, RD, and eth-tag (different GW-IP), a different `add-paths` path ID is used with each route. In Figure 29: Load balancing based on GW-IP, path IDs 1, 2 and 3 are used. This assumes `add-paths` is previously configured for the family EVPN at the default network instance.

The same behavior applies to L2, where Leaf 7 receives the same prefix 99.99.99.99/32 in six IP prefix routes and with three different GW-IPs.

Resolution of non-zero GW IP EVPN-IFL routes

The resolution of non-zero GW-IP IP prefix routes is enabled by the **evpn-interface-less-gateway-ip resolve admin-state enable** command. The following is true when **evpn-interface-less-gateway-ip resolve admin-state enable** is configured:

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

- The ingress node installs the EVPN-IFL (with non-zero GW-IP) routes (99.99.99.99 in Figure 29: Load balancing based on GW-IP) with resolution information of other routes and along with the corresponding weights.
- The router can only resolve the GW-IP to EVPN-IFL-HOST routes or a local route whose IP contains the GW-IP. A local route can be:
  - IRB interface prefix
  - ARP-ND route
- On PEs not attached to the subnet of the GW-IP, **bgp_mgr** resolves GW-IPs to EVPN-IFL-HOST routes. If these EVPN-IFL-HOST routes are received with a non-zero ESI, they are recursively resolved to the corresponding AD per-EVI routes associated with that ESI.
- For local nodes (attached to the subnet of the GW-IP), the GW-IPs are resolved to local IRB or ARP-ND routes. Local and remote next hops can be combined in a single next-hop-group and load balancing can be completed

ECMP is supported across multiple EVPN IFL routes with one of the following properties:

- non-zero GW-IP
- non-zero ESI
- zero GW-IP and ESI

If the ECMP setting is configured correctly, and three IP prefix routes are received with the preceding characteristics, an ECMP set is created that includes the next hops of all the EVPN-IFL-HOST and AD per-EVI routes that resolve the GW-IP, all the AD per-EVI routes that resolve the ESI, and the next hop of the EVPN IFL route with zero GW-IP and ESI.

In Figure 29: Load balancing based on GW-IP, Leaf 7 resolves 99.99.99.99/32 to L3 with weight=1 (because of the EVPN-IFL-HOST route for 10.2 from L3) and to L6 with weight=2 (because of the EVPN-IFL-HOST routes for 10.3 and 10.4 advertised from L6).

Static route trigger for non-zero GW IP prefix routes

Static route trigger for non-zero GW-IP prefix routes modifies GW-IP-based load balancing so that the node advertising the IP prefix route can be based on both BGP PE-CE routes and static routes. This modification is present on the transmission side, while the resolution process does not change. No CLI knob is required to enable this feature.

For example, if **evpn-interface-less-gateway-ip advertise** is enabled, and there is a BGP route, OSPF route, and static route to 100.0.0.0/24 with a next-hop 10.0.0.1, the OSPF route does not trigger the advertisement of the non-zero GW-IP. Between the BGP PE-CE and the static route, the GW-IP in the RT5 is based on the static route.

It is also possible to have ECMP enabled and have multiple static routes with different next-hops. In this case, the router advertises multiple RT5s with different GW-IPs and different path-ids (assuming **add-path** is enabled for EVPN).

Static route trigger for non-zero ESI IP prefix routes also modifies EVPN IP aliasing. For more information, see EVPN IP aliasing.

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

4.7 Dynamic multicast ID management

In SR Linux, multicast IDs (MCIDs) are multicast groups used by systems in the data path. MCIDs are allocated by the SR Linux mcid_mgr application so that the XDP layer can program different multicast forwarding modes in network instances.

For example, on 7220 IXR-D*x* platforms, the system allocates two MCIDs for each MAC-VRF network instance:

- One MCID is used for the default multicast or flood list.
- On EVPN-VXLAN MAC-VRFs, another MCID is dedicated to allow multi-homing correct forwarding of BUM traffic. Traffic coming from VXLAN that needs to be flooded automatically picks up the secondary multicast group that includes local single-homed subinterfaces plus Designated Forwarder (DF) subinterfaces, as opposed to the default flood list.
- Similarly, on EVPN-MPLS MAC-VRFs, another MCID is dedicated to BGP-EVPN instances with MPLS enabled.

The available MCIDs in SR Linux are divided among the following applications:

- net_instance_mgr, which requests an MCID per default flood list (allocated per MAC-VRF when the MAC-VRF is configured)
- evpn_mgr, which requests MCIDs per MAC-VRF (one MCID is used for EVPN multi-homing, allocated when VXLAN is configured on the MAC-VRF, and another one is used for BGP-EVPN MPLS instances)
- l2_proxy_arp_nd_mgr, which requests an MCID per MAC-VRF when the proxy-ARP/ND flood options `flood gratuitous-arp`, `unknown-arp-rq` are set to `false` to suppress flooding over EVPN destinations.
- Individual multicast modules, which request an MCID per MFIB entry created by IGMP/MLD/PIM/MVPN.

On 7250 IXR platforms, the allocation of MCIDs is similar to the allocation in 7220 IXR-D*x* platforms. When configured for multi-instance MAC-VRFs, 7250 IXR systems allocate one MCID per BGP-EVPN instance in the MAC-VRF.

Note: The use of MCIDs and the maximum number of MCIDs vary depending on the platform.

Request-response model for MCIDs

To accommodate the SR Linux features that require MCIDs, the system allocates MCIDs dynamically, using a request-response model. When an application needs a new MCID, it makes a request to the mcid_mgr application, which allocates the MCID and publishes a response with the allocated MCID to the requesting application. The mcid_mgr application also programs XDP with the MCID.

MCID exhaustion

The system has a finite number of MCIDs to allocate. If mcid_mgr runs out of MCIDs it can allocate, the following applications are brought down when a new object is configured:

- mac-vrf
- vxlan-interface
- bgp-evpn mpls instance

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

When this occurs, the `oper-down-reason` for the application is displayed as `no-mcid`. When the `oper-state` for an application is changed to `down`, then brought back up, if there are no MCIDs available, the application is kept down with `oper-down-reason no-mcid`. For example, a change in `evi` automatically toggles the `oper-state` of an EVPN instance. If there are no MCIDs available following the `evi` change, the EVPN instance may not come back up.

Upon MCID exhaustion, mcid_mgr keeps trying to allocate MCIDs for the requesting multicast applications. If the system is rebooted, it may result in some VXLAN interfaces / MAC-VRFs that were not allocated MCIDs earlier receiving them at the expense of the multicast routes.

For l2_proxy_arp_nd_mgr, the application is brought down only if the flood options that require a new MCID are configured.

4.7.1 Displaying MCID information

Procedure

You can use an **info from state** command to display information about MCID allocation and MCID usage per application.

The command displays the total number of MCIDs available in the system, the total number of MCIDs currently in use, and the total number of pending MCIDs that mcid_mgr has not yet allocated.

For each application that requires MCIDs (Layer 2 proxy ARP/ND, MAC VRFs, MAC VRF BGP-EVPN, and VXLAN interfaces) you can display the number of MCIDs in use by the application, as well as the number of MCIDs requested but not yet allocated for use by the application.

Example

The following command displays the current MCID usage:

```
--{ running }--[  ]--
# info from state with-context system multicast multicast-ids statistics
    system {
        multicast {
            multicast-ids {
                statistics {
                    maximum-ids 16380
                    current-usage 3
                    total-pending 0
                    multicast-id-user-type mac-vrf {
                        current-usage 2
                        total-pending 0
                    }
                    multicast-id-user-type vxlan-interface {
                        current-usage 1
                        total-pending 0
                    }
                    multicast-id-user-type l2-proxy-arp-nd {
                        current-usage 0
                        total-pending 0
                    }
                    multicast-id-user-type mfib {
                        current-usage 0
                        total-pending 0
                    }
                    multicast-id-user-type mac-vrf-bgp-evpn {
                        current-usage 0
                        total-pending 0
```


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

```
                    }
                }
            }
        }
    }
```

4.8 EVPN for multicast

This chapter describes EVPN for multicast on SR Linux.

Note: The components of EVPN for multicast are currently supported on 7220 IXR-D2/D3/D2L/D3L/D4/D5 and 7250 IXR Gen 2c+ platforms.

4.8.1 IGMP/MLD snooping

Note: IGMP/MLD snooping are supported on 7220 IXR-D2/D2L/D3/D3L/D4/D5 and 7250 IXR Gen 2c+ platforms.

IGMP snooping is enabled by configuring **igmp-snooping admin-state enable** under **network-instance protocols**. This feature is supported on MAC-VRF network instances with bridged subinterfaces.

The `igmp-snooping` model allows configuration of IGMP parameters at the MAC-VRF network instance level and specific configuration for individual interfaces. The programming of the MFIB for the network-instance is enabled by the configuration of **network-instance protocols igmp-snooping**.

IGMP snooping support in SR Linux includes the following:

- IGMP versions 1, 2, and 3 are supported.
- IGMP snooping is supported on network instances of type **mac-vrf**.
- The following features are supported:
  - Querier support, including timers and relevant parameters
  - Trace options
  - Mcast router support
  - State for group memberships, queriers, and mcast routers
  - Static membership groups configured under interfaces
  - Statistics
- As soon as IGMP snooping is enabled on the MAC-VRF, the node starts snooping IGMP messages on the interfaces added under `protocols igmp-snooping`, and starts programming the MFIB and proxying IGMP reports.
  The interfaces not configured under `protocols igmp-snooping` but that exist in the network instance are considered mrouter ports by the IGMP snooping application. The state model shows the learned queriers, multicast routers, and proxy membership groups.
- IGMP settings can be customized at the interface level. The state model shows the learned membership groups on the interface, as well as the statistics.
- When originating reports from CPM, the system MAC is used.

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

MLD snooping is modeled after IGMP snooping (both are managed by IGMP manager), with MLD versions 1 and 2 supported. Similar to IGMP snooping, MLD snooping is enabled with the command **mld-snooping admin-state enable**.

4.8.2 EVPN multicast for Layer 2

This section describes EVPN multicast for Layer 2 on SR Linux.

IGMP/MLD proxies for EVPN (RFC 9251)

SR Linux supports IGMP/MLD proxies for EVPN networks, as described in RFC 9251.

When IGMP snooping and, or MLD snooping are enabled on a MAC-VRF, a multicast forwarding information base (MFIB) is created with entries for each (*,G) or (S,G) state that is snooped out of received IGMP/MLD reports. The result of that MFIB is advertised in EVPN Selective Multicast Ethernet Tag (SMET) routes, one route per (*,G)/(S,G) entry, as long as EVPN is enabled in the MAC-VRF.

The following summarizes what happens when both `igmp/mld-snooping` and `bgp-evpn` are enabled:

- The router sends updates of the IMET routes for the MAC-VRF with the multicast flags EC signaling support for IGMP/MLD-proxy.
- The router collects information about the remote EVPN destinations that support iIGMP/MLD-proxy (based on the received IMET flags) and stops flooding IGMP reports to those destinations.
- The router also creates an EVPN proxy database and sends the result in SMET routes.
- The router adds EVPN destinations as OIF (outgoing interfaces) in the MFIB based on the received SMET routes.

The SMET route (or EVPN route type 6) is described in RFC 9251 and conveys the information about the multicast source and the multicast group IP addresses (for IPv4 or IPv6) along with some flags. The flags indicate whether the proxy entry in the MFIB was created using IGMP/MLDv2 or IGMPv3; in case of IGMPv3/MLDv2, the `Exclude` flag (when set to 1) indicates if the source is excluded.

On reception of an SMET route, IGMP/MLD reports are generated and sent to mrouter subinterfaces based on the IGMP/MLD version that is locally valid.

Integration of EVPN proxy and non-proxy routers

SR Linux supports the integration of EVPN multicast proxy-enabled and non-proxy routers in the same EVPN broadcast domain. As described in RFC 9251, each router where IGMP/MLD snooping is enabled advertises an IMET route along with the Multicast Flags EC. This EC is used by the router to indicate that it has enabled IGMP/MLD snooping for that MAC-VRF.

Upon reception of the IMET route, a PE considers that the originating PE supports the EVPN proxy behavior if and only if, the Multicast Flags EC is present *and* the proxy bit value is 1, else the originating PE is not considered as an EVPN proxy PE. This information is used for backwards compatibility with non-supporting proxy or non-upgraded PEs. The following considerations apply:

- On the egress PE (connected to receivers): If an egress PE supports EVPN multicast proxy, but does not have any interest in a given (x, G) (with x meaning source S or *), it advertises its IGMP proxy capability using this EC, but does not advertise any SMET route for that (x, G).
- On the ingress PE (connected to sources):
  - When the PE receives an IMET advertisement with proxy support from the egress PE, it does not replicate the multicast traffic to that PE until a SMET route is received from it.

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

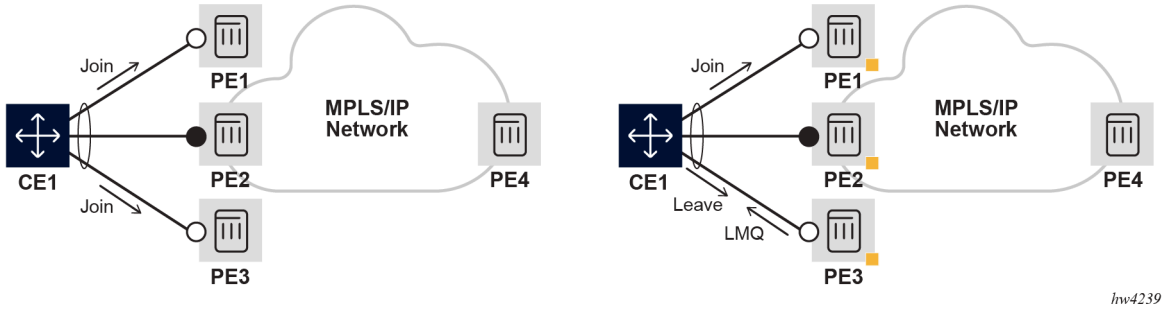
– When the PE receives an IMET route with *no* proxy support from an egress PE, multicast traffic is replicated to that egress PE, even if the egress PE has no receivers for the stream. The egress PE is in this case added as an OIF for (*,*); that is, multicast is sent to it for any sources and groups.
– Reports and queries are still forwarded to non-proxy PEs and processed on reception from non-proxy PEs.

To summarize, an EVPN multicast proxy PE never sends IGMP/MLD reports to EVPN destinations created from IMET routes with EVPN multicast proxy support, but it does send reports to EVPN destinations created from IMET routes without the proxy indication. This behavior allows the interworking between EVPN multicast proxy and non-proxy PEs.

4.8.2.1 EVPN multihoming and multicast synchronization

As described in RFC 9251, on EVPN multicast proxy-enabled PEs, multicast states must be synchronized across all the PEs attached to the same ES. This is illustrated in the following figure:

Figure 30: Multicast state synchronization for EVPN MH



The left side of the figure shows how it is possible for CE1 to send Join messages for a (S,G) to PE1 or PE3, while PE2 is the DF and therefore the only PE that can send multicast traffic to the CE. The multicast traffic would not be delivered to the CE1 unless the DF, PE2, creates a state for the (S,G).

The right side of the figure shows a similar issue for the Leave process. CE1 may send a Leave message to PE3, which replies back with a LMQ (Last Member Query). CE1 then may send a Join message to a different PE as a result of a different hashing; for example, PE1. Therefore, PE3 would think the CE has left the group and would remove the state.

These two issues are solved by synchronizing the multicast states in all the PEs attached to the same ES. This feature introduces two new EVPN route types described in RFC 9251 to synchronize the multicast states in EVPN MAC-VRFs:

- Route type 7, or Multicast Join Synch Route (synchronizes the join state in all the PEs of the ES).
- Route type 8, or Multicast Leave Synch Route (synchronizes the leave state in all the PEs of the ES).

Note that this feature is needed along with all-active and single-active multi-homing. In single-active multi-homing, the synchronization helps speed up convergence.

Operation of EVPN multicast and multihoming

EVPN multicast synchronization in multi-homed PEs does not require any specific configuration command. Configuration of **igmp/mld-snooping admin-state enable** and the existence of multicast state received

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

in a bridged subinterface attached to an ES is the trigger for the router to advertise the EVPN multicast synchronization routes.

The following considerations apply:

- Reception of IGMP/MLD Join and Leave messages on a local ES triggers the advertisement of EVPN multicast Join/Leave synchronization routes respectively and the associated procedures in RFC 9251. These routes are advertised only if ES bridged subinterfaces with multicast state exist.

  The routes are always advertised with the ESI-import route-target (which ensures the routes are only imported by the PEs attached to the ES) and the EVI route target EC (which guarantees the routes are only imported in the correct MAC-VRF).
- Reception of multicast Join/Leave synchronization routes trigger the synchronization of states and the associated procedures described in RFC 9251.
- About the Leave synchronization routes:
  - Leave synchronization routes synchronize the LMQ (Last Member Query) procedures in the ES peers.
  - The node triggering the route encodes (Last Member Query Interval * robust count) in the Maximum Response Time field of the NLRI, but applies an extra delta before expiring the max response time locally. This delta time is configurable with **system.network-instance.protocols.evpn.multicast.leave-sync-propagation <number>**, with a 0-to-300 second range and default value of 5 seconds.
  - This delta is a configured value, ideally configured the same on all the ES peers and long enough to account for the BGP propagation time between the ES peers.
    For example, suppose there is a max response time of 5 seconds (50 deciseconds), advertised by A and a delay of 4 seconds in BGP propagation to B. In this case, the timer could already expire on A while B is still in LMQ time and can still receive Joins (which would recreate state in A after a Join synchronization route from B).

    By adding an extra delta on A, it minimizes the potential churn of A removing and recreating the state.
  - The advertised maximum response time value in the multicast Leave synchronization route is the configured `robust-count * query-last-member-interval`, by default 2 seconds. The `robust-count` is taken from the `igmp` context (the value is defined by the querier), and the `query-last-member-interval` is taken from the ES interface `igmp-snooping` configuration.
  - Fast Leave is supported along with the EVPN Leave synchronization routes. In this case, the EVPN Leave synchronization route is advertised with a Max Response Time of 0 seconds.

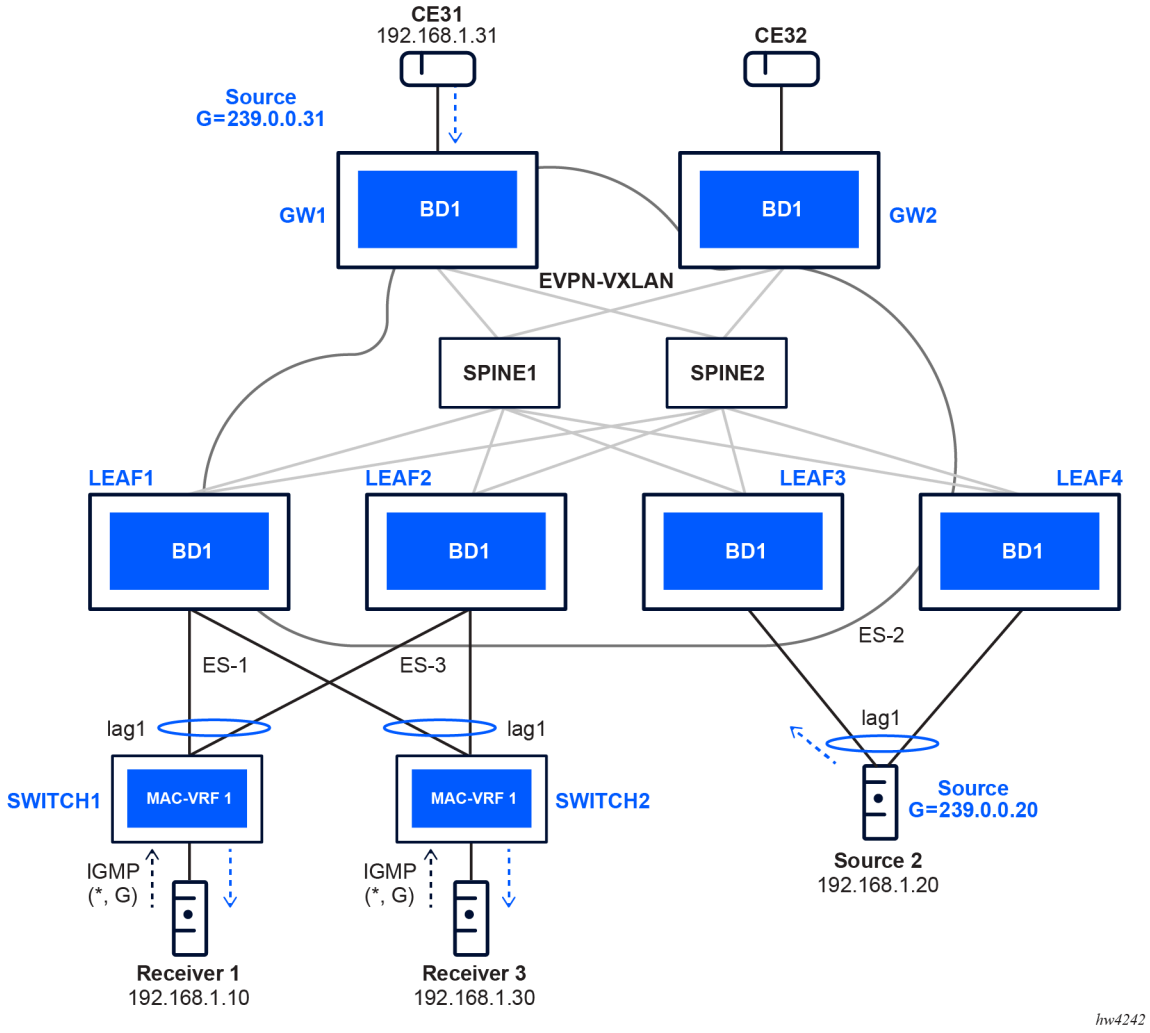
4.8.2.2 Configuring EVPN multicast for Layer 2

To configure EVPN multicast for Layer 2, **igmp/mld-snooping** must be enabled on the MAC-VRF.

The following example illustrates how EVPN multicast for Layer 2 is configured.

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

Figure 31: EVPN multicast for Layer 2



Where:

- GW1 and GW2 are two data center gateways (DC GWs).
- LEAF1 through LEAF4 are four SR Linux nodes. LEAF3 and LEAF4 are multi-homed to Client 2, a source that generates traffic for group 239.0.0.20.
- The four leaf nodes and the two DC gateway nodes are attached to the same EVPN BD.
- SWITCH1 and SWITCH2 are two SR Linux systems that are not EVPN-enabled, but have IGMP snooping enables so they can generate IGMP reports on behalf of the clients. The two switches are dual-homed to LEAF1 and LEAF2, using all-active Ethernet Segments.

The configuration of the MAC-VRF in LEAF1 and its Ethernet Segments is as follows. The other leaf nodes are configured in a similar way.

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance "MAC-VRF 1"
```


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

```
    network-instance "MAC-VRF 1" {
        type mac-vrf
        interface ethernet-1/1.1 {
        }
        interface ethernet-1/2.1 {
        }
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan0.1
                    evi 1
                    ecmp 2
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                    route-distinguisher {
                        rd 1:11
                    }
                    route-target {
                        export-rt target:65011:1
                        import-rt target:65011:1
                    }
                }
            }
            igmp-snooping {  // this is the configuration required for EVPN multicast or evpn-
proxy in SRL
                admin-state enable
                trace-options {
                    trace {
                        packet {
                            modifier egress-ingress-and-dropped
                        }
                    }
                }
                interface ethernet-1/1.1 {
                    version 2
                    send-queries true  // queries are generated locally since queries are not
 propagated in EVPN
                }
                interface ethernet-1/2.1 {
                    version 2
                    send-queries true
                }
            }
        }
    }
```

```
--{ * candidate shared default }--[  ]--
# info with-context system network-instance
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment ES-SROS-Client1 {
                                admin-state enable
                                esi 01:01:01:01:01:01:01:01:01:01
                                multi-homing-mode all-active
```


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

```
                                interface ethernet-1/1 {
                                }
                            }
                            ethernet-segment ES-SROS-Client3 {
                                admin-state enable
                                esi 03:03:03:03:03:03:03:03:03:03
                                multi-homing-mode all-active
                                interface ethernet-1/2 {
                                }
                            }
                        }
                    }
                }
                bgp-vpn {
                    bgp-instance 1 {
                    }
                }
            }
        }
    }
```

Note that **send-queries true** must be configured on the interfaces of the MAC-VRF. This is because of the leaf routers not flooding the queries from the multicast routers. By enabling **send-queries** on all the interfaces, the EVPN BD becomes a virtual multicast router to the receivers. This is as described in RFC 9251.

Also note that trace-options are enabled to facilitate troubleshooting for the multicast messages.

Configuration of SWITCH1 and SWITCH2 is simpler:

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance "MAC-VRF 1"
    network-instance "MAC-VRF 1" {
        type mac-vrf
        interface ethernet-1/1.1 {
        }
        interface lag1.1 {
        }
        protocols {
            igmp-snooping {
                admin-state enable
                trace-options {
                    trace {
                        packet {
                            modifier egress-ingress-and-dropped
                        }
                    }
                }
                interface ethernet-1/1.1 {
                    static-membership-groups {
                        group 239.0.0.20 {
                            starg
                        }
                        group 239.0.0.31 {
                            source 192.168.1.31 {
                            }
                        }
                    }
                }
                interface lag1.1 {
                }
            }
        }
```


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

```
    }
```

4.8.2.3 Displaying EVPN multicast for Layer 2 information

The examples in this section display multicast information for the LEAF routers in the example configuration in Figure 31: EVPN multicast for Layer 2 .

The following example shows the MFIB table in MAC-VRF 1:

```
--{ running }--[  ]--
# info from state with-context network-instance "MAC-VRF 1" multicast-forwarding-information-
base multicast-route 0.0.0.0 group 239.0.0.20
    network-instance "MAC-VRF 1" {
        multicast-forwarding-information-base {
            multicast-route 0.0.0.0 group 239.0.0.20 {
                last-update "2024-01-11T02:52:39.241Z (4 days ago)"
                outgoing-next-hop-group 203145816567 {
                    forward true
                }
                outgoing-next-hop-group 203145816569 {
                    forward true
                }
                outgoing-next-hop-group 203145816575 {
                    forward true
                }
            }
        }
    }
```

You can also use the **show network-instance multicast-forwarding-information-base** command to display the MFIB; for example:

```
--{ candidate shared default }--[ network-instance "MAC-VRF 1" ]--
A:leaf1# show multicast-forwarding-information-base
-------------------------------------------------------------------------------
IPv4 multicast FIB of network instance MAC-VRF 1
-------------------------------------------------------------------------------
+---------+------------+-------------------+---------+-------------+-------------+
| Source  | Group      | OIF list          | Forward | Line card   | Last Update |
|         |            |                   |         | Replication |             |
|         |            |                   |         | Index       |             |
+=========+============+===================+=========+=============+=============+
| 0.0.0.0 | 0.0.0.0    |                   |         | 1           | 5h52m8s ago |
| 0.0.0.0 | 239.0.0.20 | ethernet-1/1.1    | True    | 2           | 5h52m6s ago |
|         |            | ethernet-1/2.1    | True    |             |             |
|         |            | vxlan:10.0.0.12:1 | True    |             |             |
|         |            |                   |         |             |             |
| 0.0.0.0 | 239.0.0.31 | ethernet-1/1.1    | True    | 3           | 5h52m4s ago |
|         |            | ethernet-1/2.1    | True    |             |             |
|         |            | vxlan:10.0.0.12:1 | True    |             |             |
|         |            |                   |         |             |             |
+---------+------------+-------------------+---------+-------------+-------------+
-------------------------------------------------------------------------------
IPv4 routes total : 3
IPv6 routes total : 0
-------------------------------------------------------------------------------
```


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

The following command shows that all remote VTEPs support the EVPN multicast proxy features:

```
--{ candidate shared default }--[ ]--
A:leaf3# show network-instance "MAC-VRF 1" protocols igmp-snooping vxlan-destination
=============================================-======================
Net-Inst "MAC-VRF 1" IGMP SNOOPING vxlan destinations
=====================================================================
+-----------+-----+-------------+------------+---------------+--------+--------+
| Vtep      | Vni | NHG Id      | Is MRouter | Is Evpn Proxy | Is SBD | Groups |
+===========+=====+=============+============+===============+========+========+
| 10.0.0.11 | 1   | 203145816567 | false      | true          | false  | 2      |
| 10.0.0.12 | 1   | 203145816569 | false      | true          | false  | 2      |
| 10.0.0.14 | 1   | 203145816565 | false      | true          | false  | 0      |
| 10.0.0.31 | 1   | 203145816575 | false      | true          | false  | 1      |
| 10.0.0.32 | 1   | 203145816573 | false      | true          | false  | 0      |
+-----------+-----+-------------+------------+---------------+--------+--------+
---------------------------------------------------------------------------------
No. of destinations: 5
=================================================================================
```

The IGMP snooping state on LEAF1 is as follows:

```
--{ candidate shared default }--[ ]--
A:leaf1# info from state with-context network-instance "MAC-VRF 1" protocols igmp-snooping
    network-instance "MAC-VRF 1" {
        protocols {
            igmp-snooping {
                admin-state enable
                query-interval 125
                robust-count 2
                oper-state up
                transmitted-bgp-smet-routes 22
                proxy-membership-group-count 2
                proxy-evpn-membership-group-count 2
                trace-options {
                    trace {
                        packet {
                            modifier egress-ingress-and-dropped
                        }
                    }
                }
                proxy-membership-groups {  // --> proxy DB
                    group 239.0.0.20 {
                        filter-mode exclude
                        up-time "2024-01-11T02:52:38.000Z (4 days ago)"
                    }
                    group 239.0.0.31 {
                        filter-mode exclude
                        up-time "2024-01-11T02:52:39.000Z (4 days ago)"
                    }
                }
                proxy-evpn-membership-groups { // --> proxy-evpn DB used to propagate the reports with
 the proper flags
                    group 239.0.0.20 {
                        filter-mode exclude
                        up-time "2024-01-11T02:52:38.000Z (4 days ago)"
                        v1-support false
                        v2-support true
                        v3-support false
                    }
                    group 239.0.0.31 {
                        filter-mode exclude
                        up-time "2024-01-11T02:52:39.000Z (4 days ago)"
```


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

```
                        v1-support false
                        v2-support true
                        v3-support false
                    }
                }
                interface ethernet-1/1.1 {
                    router-alert-check true
                    version 2
                    maximum-number-groups 0
                    maximum-number-sources 0
                    maximum-number-group-sources 0
                    query-interval 125
                    query-last-member-interval 1
                    query-response-interval 10
                    robust-count 2
                    fast-leave false
                    mrouter-port false
                    send-queries true
                    membership-group-count 2
                    is-mrouter-port false
                    membership-groups {
                        group 239.0.0.20 {
                            group-type dynamic // --> type is dynamic because the report from SWITCH1
is hashed to LEAF1. On LEAF2, it shows as "group-type bgp-sync"
                            filter-mode exclude
                            expiry-time 245
                            up-time "2024-01-11T02:52:38.000Z (4 days ago)"
                            v1-host-timer 0
                            v2-host-timer 245
                            igmp-compatibility-mode 2
                        }
                        group 239.0.0.31 {
                            group-type dynamic
                            filter-mode exclude
                            expiry-time 242
                            up-time "2024-01-11T02:52:39.000Z (4 days ago)"
                            v1-host-timer 0
                            v2-host-timer 242
                            igmp-compatibility-mode 2
                        }
                    }
                    statistics {
                        received {
                            general-queries 0
                            group-queries 0
                            group-source-queries 0
                            v1-reports 0
                            v2-reports 5860
                            v3-reports 6
                            leave-messages 3
                            discarded-packets 2
                            bgp-join-sync 4
                            bgp-leave-sync 0
                        }
                        transmitted {
                            general-queries 2
                            group-queries 11
                            group-source-queries 0
                            v1-reports 0
                            v2-reports 0
                            v3-reports 0
                            leave-messages 0
                            error-packets 0
                            bgp-join-sync 6
```


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

```
                        bgp-leave-sync 6
                    }
                    forwarded {
                        general-queries 0
                        group-queries 0
                        group-source-queries 0
                        v1-reports 0
                        v2-reports 0
                        v3-reports 0
                        leave-messages 0
                        unknown-type 0
                        error-packets 0
                    }
                    error {
                        bad-length 0
                        unknown-type 0
                        wrong-version 0
                        missing-router-alert 0
                        bad-encoding 0
                        local-scope 0
                        reached-maximum-number-groups 0
                        reached-maximum-number-group-sources 0
                        reached-maximum-number-sources 0
                        out-of-memory-discarded-packets 0
                        bad-igmp-checksum 0
                        zero-source-ip-address 0
                        send-query-configured-discarded-packets 0
                        discarded-bgp-join-sync 0
                        discarded-bgp-leave-sync 0
                    }
                    multicast-states {
                        star-group-entries 2
                        source-group-entries 0
                    }
                }
            }
            interface ethernet-1/2.1 {
                router-alert-check true
                version 2
                maximum-number-groups 0
                maximum-number-sources 0
                maximum-number-group-sources 0
                query-interval 125
                query-last-member-interval 1
                query-response-interval 10
                robust-count 2
                fast-leave false
                mrouter-port false
                send-queries true
                membership-group-count 2
                is-mrouter-port false
                membership-groups {
                    group 239.0.0.20 {
                        group-type dynamic
                        filter-mode exclude
                        expiry-time 246
                        up-time "2024-01-11T02:54:43.000Z (4 days ago)"
                        v1-host-timer 0
                        v2-host-timer 246
                        igmp-compatibility-mode 2
                    }
                    group 239.0.0.31 {
                        group-type dynamic
                        filter-mode exclude
```


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

```
                        expiry-time 249
                        up-time "2024-01-11T02:54:44.000Z (4 days ago)"
                        v1-host-timer 0
                        v2-host-timer 249
                        igmp-compatibility-mode 2
                    }
                }
                statistics {
                    received {
                        general-queries 0
                        group-queries 0
                        group-source-queries 0
                        v1-reports 0
                        v2-reports 5858
                        v3-reports 0
                        leave-messages 0
                        discarded-packets 0
                        bgp-join-sync 0
                        bgp-leave-sync 0
                    }
                    transmitted {
                        general-queries 2
                        group-queries 0
                        group-source-queries 0
                        v1-reports 0
                        v2-reports 0
                        v3-reports 0
                        leave-messages 0
                        error-packets 0
                        bgp-join-sync 2
                        bgp-leave-sync 0
                    }
                    forwarded {
                        general-queries 0
                        group-queries 0
                        group-source-queries 0
                        v1-reports 0
                        v2-reports 0
                        v3-reports 0
                        leave-messages 0
                        unknown-type 0
                        error-packets 0
                    }
                    error {
                        bad-length 0
                        unknown-type 0
                        wrong-version 0
                        missing-router-alert 0
                        bad-encoding 0
                        local-scope 0
                        reached-maximum-number-groups 0
                        reached-maximum-number-group-sources 0
                        reached-maximum-number-sources 0
                        out-of-memory-discarded-packets 0
                        bad-igmp-checksum 0
                        zero-source-ip-address 0
                        send-query-configured-discarded-packets 0
                        discarded-bgp-join-sync 0
                        discarded-bgp-leave-sync 0
                    }
                    multicast-states {
                        star-group-entries 2
                        source-group-entries 0
                    }
```


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

```
                }
            }
            vxlan-destination 10.0.0.12 vni 1 {
                index 203145159966
                is-mrouter-port false
                is-evpn-proxy true
                is-sbd false
                membership-group-count 2
                membership-groups {
                    group 239.0.0.20 {
                        group-type bgp-smet  // --> created out of an SMET route
                        filter-mode exclude
                        expiry-time 2147483647
                        up-time "2024-01-11T02:52:40.000Z (4 days ago)"
                        v1-host-timer 0
                        v2-host-timer 0
                        igmp-compatibility-mode 2
                    }
                    group 239.0.0.31 {
                        group-type bgp-smet
                        filter-mode exclude
                        expiry-time 2147483647
                        up-time "2024-01-11T02:52:41.000Z (4 days ago)"
                        v1-host-timer 0
                        v2-host-timer 0
                        igmp-compatibility-mode 2
                    }
                }
                statistics {
                    received-smet 10
                    discarded-smet 0
                }
            }
            vxlan-destination 10.0.0.13 vni 1 {
                index 203145159969
                is-mrouter-port false
                is-evpn-proxy true
                is-sbd false
                membership-group-count 0
                statistics {
                    received-smet 0
                    discarded-smet 0
                }
            }
            vxlan-destination 10.0.0.14 vni 1 {
                index 203145159964
                is-mrouter-port false
                is-evpn-proxy true
                is-sbd false
                membership-group-count 0
                statistics {
                    received-smet 0
                    discarded-smet 0
                }
            }
            vxlan-destination 10.0.0.31 vni 1 {
                index 203145159974
                is-mrouter-port false
                is-evpn-proxy true
                is-sbd false
                membership-group-count 1
                membership-groups {
                    group 239.0.0.20 {
                        group-type bgp-smet
```


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

```
                            filter-mode exclude
                            expiry-time 2147483647
                            up-time "2024-01-11T07:52:24.000Z (4 days ago)"
                            v1-host-timer 0
                            v2-host-timer 0
                            igmp-compatibility-mode 2
                        }
                    }
                    statistics {
                        received-smet 4
                        discarded-smet 0
                    }
                }
                vxlan-destination 10.0.0.32 vni 1 {
                    index 203145159972
                    is-mrouter-port false
                    is-evpn-proxy true
                    is-sbd false
                    membership-group-count 0
                    statistics {
                        received-smet 0
                        discarded-smet 0
                    }
                }
            }
        }
    }
```

Configuring the **trace-options** command allows you to obtain detailed logs. For example:

```
2024-01-15T08:40:59.875930+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11802|D: VR 10240 IGMP
 RX packet on network-instance "MAC-VRF 1":
  SrcMac    : 1a:61:0c:ff:00:00
  Interface : ethernet-1/2.1
  SrcIp     : 0.0.0.0
  DstIp     : 239.0.0.31
  Type      : V2 REPORT
    GroupAddr   : 239.0.0.31
2024-01-15T08:41:19.798150+00:00 leaf1 local6|INFO sr_cli: debug|29762|29762|00024|I: common    |root|
1007|srl candidate shared default / | bash viewlog -e igmp -t
2024-01-15T08:42:48.828291+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11803|D: VR 10240 IGMP
 RX packet on network-instance "MAC-VRF 1":
  SrcMac    : 1a:9a:0b:ff:00:00
  Interface : ethernet-1/1.1
  SrcIp     : 0.0.0.0
  DstIp     : 224.0.0.2
  Type      : V2 LEAVE
    GroupAddr   : 239.0.0.20
2024-01-15T08:42:48.829266+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11804|D: VR 10240 IGMP
 TX packet on network-instance "MAC-VRF 1":
  DstMac    : 01:00:5e:00:00:14
  Interface : ethernet-1/1.1
  SrcIp     : 10.0.0.11
  DstIp     : 239.0.0.20
  Type      : QUERY
    MaxRespCode : 10 (0xa)
    GroupAddr   : 239.0.0.20
2024-01-15T08:42:48.829737+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11805|D: VR 10240 IGMP
 TX BGP MCAST_LEAVE_SYNCH Route on network-instance "MAC-VRF 1":   from local port 1/1/1
  ADD (*,239.0.0.20) V2 maxRespTime 20
2024-01-15T08:42:49.812651+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11806|D: VR 10240 IGMP
 TX packet on network-instance "MAC-VRF 1":
  DstMac    : 01:00:5e:00:00:14
```


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

```
  Interface : ethernet-1/1.1
  SrcIp     : 10.0.0.11
  DstIp     : 239.0.0.20
  Type      : QUERY
    MaxRespCode : 10 (0xa)
    GroupAddr   : 239.0.0.20
2024-01-15T08:42:55.825134+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11807|D: VR 10240 IGMP
 TX BGP MCAST_LEAVE_SYNCH Route on network-instance "MAC-VRF 1":   from local port 1/1/1
  DEL (*,239.0.0.20) maxRespTime 0
2024-01-15T08:42:55.825283+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11808|D: VR 10240 IGMP
 TX BGP MCAST_JOIN_SYNCH Route on network-instance "MAC-VRF 1":   from local port 1/1/1
  DEL (*,239.0.0.20)
2024-01-15T08:43:00.044934+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11809|D: VR 10240 IGMP
 RX packet on network-instance "MAC-VRF 1":
  SrcMac    : 1a:9a:0b:ff:00:00
  Interface : ethernet-1/1.1
  SrcIp     : 0.0.0.0
  DstIp     : 239.0.0.31
  Type      : V2 REPORT
    GroupAddr   : 239.0.0.31
2024-01-15T08:43:00.492810+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11810|D: VR 10240 IGMP
 RX packet on network-instance "MAC-VRF 1":
  SrcMac    : 1a:61:0c:ff:00:00
  Interface : ethernet-1/2.1
  SrcIp     : 0.0.0.0
  DstIp     : 239.0.0.20
  Type      : V2 REPORT
    GroupAddr   : 239.0.0.20
2024-01-15T08:43:03.196847+00:00 leaf1 local6|DEBU sr_igmp_mgr: igmp|2532|2733|11811|D: VR 10240 IGMP
 RX packet on network-instance "MAC-VRF 1":
  SrcMac    : 1a:61:0c:ff:00:00
  Interface : ethernet-1/2.1
  SrcIp     : 0.0.0.0
  DstIp     : 239.0.0.31
  Type      : V2 REPORT
    GroupAddr   : 239.0.0.31
```

4.8.3 EVPN multicast for Layer 3 (RFC 9625)

This section describes EVPN multicast for Layer 3 on SR Linux.

Optimized Inter-Subnet Multicast (OISM) forwarding is an EVPN-based solution for the forwarding of Layer 2 and Layer 3 overlay multicast traffic from sources to receivers. The EVPN OISM data path requires the support for a supplementary broadcast domain (SBD) attached to the IP-VRFs so that leaf nodes can receive traffic if they are not attached to the source broadcast domain (BD).

Terminology

EVPN OISM uses the following terminology:

- broadcast domain (BD) – equivalent to a MAC-VRF
- ordinary BD – a BD that is not a SBD
- source BD – a BD that is connected to a source
- supplementary broadcast domain (SBD) – a BD that does not have any attachment circuits, only IRB interfaces, and is used to provide connectivity in the IP-VRF that it is attached to
- PEG – PIM to EVPN (OISM) gateway function
- MEG – MVPN to EVPN (OISM) gateway function

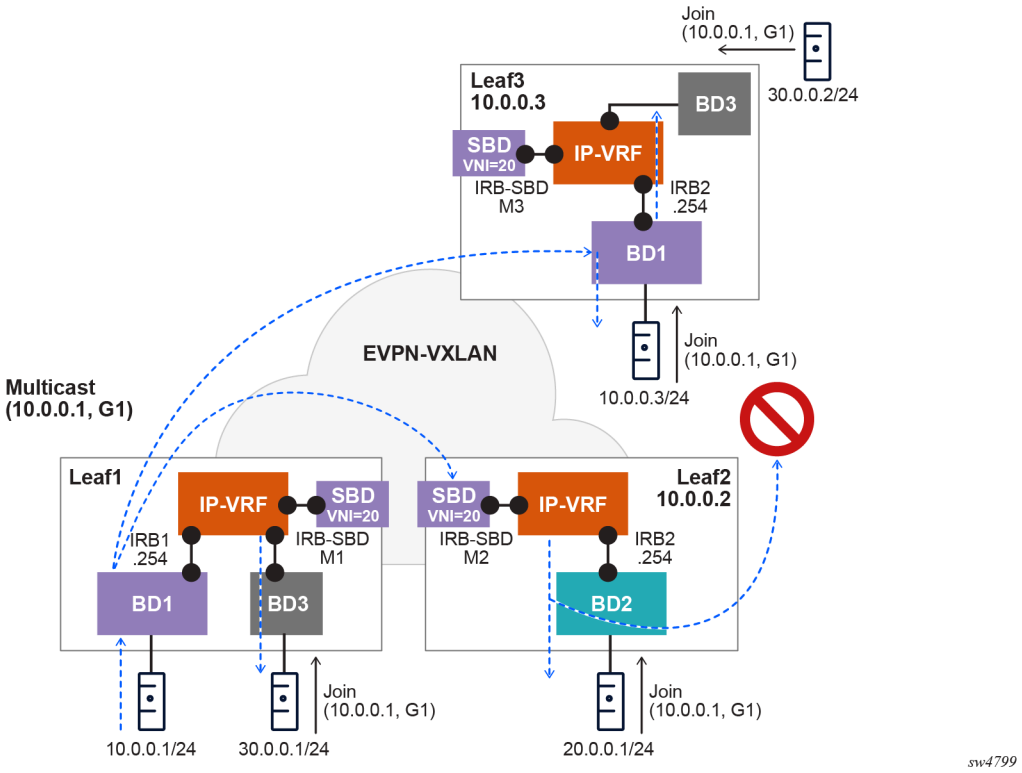
© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

- IIF – incoming interface
- OIF – outgoing interface
- SMET route – Selective Multicast Ethernet Tag route (EVPN route type 6)

4.8.3.1 Data path requirements

The following figure illustrates the OISM data path perspective.

Figure 32: OISM data path perspective



A leaf receiving multicast on a local bridged subinterface must forward multicast traffic directly to local receivers and through the source BD to remote receivers, but never to EVPN binds in a non-source BD. For example, Leaf1 sends multicast to Leaf3/Leaf2 using VXLAN binds in Leaf1 BD1 to Leaf3 BD1 and Leaf2 SBD. However, traffic locally routed by Leaf1 to BD3 is not sent again to remote leaf nodes using VXLAN binds in BD3.

A PE receiving multicast from EVPN on the source BD must forward the traffic directly to local receivers that exist in the source or different BD, but never to other EVPN destinations on the source BD or different BD. For example, Leaf3 receives traffic on the source BD (BD1), and forwards the traffic to local receivers on the same BD1 or BD3, but never to an EVPN destination.

A PE receiving multicast from EVPN on the SBD must forward the traffic to local receivers through the SBD interface and never to EVPN destinations. For example, Leaf2 receives multicast on SBD and forwards the traffic to BD2 bridge subinterfaces but not to the BD2 VXLAN bids.

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

4.8.3.2 Control plane requirements

OISM uses the Selective Multicast Ethernet Tag (SMET) route or route type 6 to signal interest on a specific device. In terms of signaling interest for a multicast group, the OISM solution supports the following:

- IGMP or MLD from the receivers
- static IGMP/MLD multicast groups, for receivers that do not support either IGMP/MLD
- BGP-EVPN SMET routes from the PEs to indicate interest to remote EVPN PEs

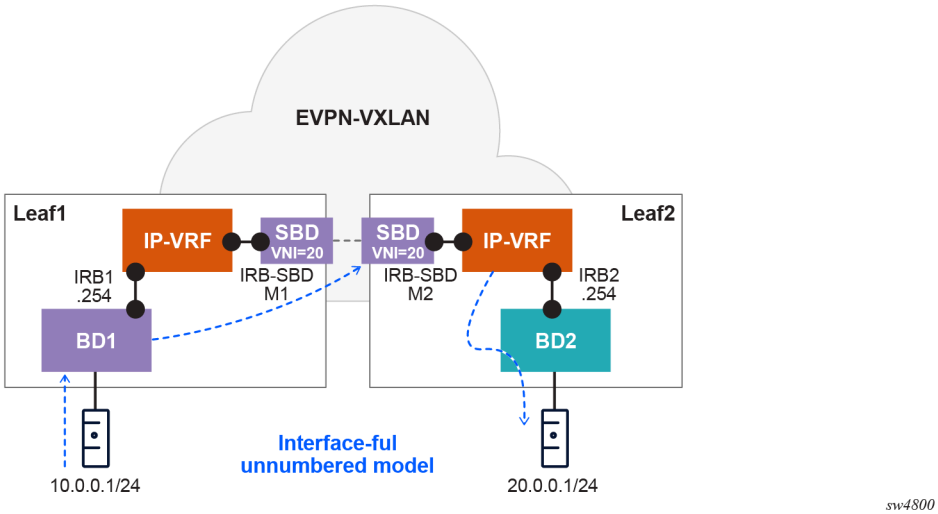
The routers build their IIF and OIF lists based on the information received on either IGMP/MLD or SMET routes.

EVPN multi-homing is also supported for multicast sources or receivers attached to a BD. When receivers are connected through all-active Ethernet Segments, the multicast state in the MFIBs is synchronized across all nodes in the Ethernet Segment using EVPN multicast synchronization routes (Type 7 and Type 8).

4.8.3.3 Configuring EVPN multicast for Layer 3

This section shows a configuration example of an OISM leaf node, as shown in the following figure:

Figure 33: EVPN OISM example



OISM is enabled by the configuration of the following:

- EVPN interface-ful unnumbered configuration model for the IP-VRF, SBD and ordinary BDs for unicast routing. The local and EVPN-IFF routes in the IP-VRF route table are used to check the multicast traffic source addresses and drop the traffic if received over an unexpected IIF. This is referred to as Reverse Path Forwarding check.
- IGMP/MLD snooping on the SBD, as well as the BDs connected to sources and receivers (BD2 in Figure 33: EVPN OISM example):

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

  - On the SBD, IGMP/MLD snooping allows for state creation across BDs, and the advertisement of SMET routes to draw the required multicast traffic.
  - On ordinary BDs, IGMP/MLD snooping is needed so that multicast can be forwarded to EVPN destinations (SBD and non-SBD destinations).
  - IGMP/MLD snooping on IRB interfaces is supported, however only the **mrouter** configuration can be modified for IRBs.
- Layer 3 IGMP/MLD on the BDs attached to sources and receivers, where IGMP/MLD is needed on the (non-SBD) IRBs so that joins can be programmed in `pim_mgr`.
- PIM in the IP-VRF, including the SBD and BD IRB interfaces attached to sources and receivers:
  - This is needed so that source addresses are resolved.
  - The SBD IRB is unnumbered, so the router needs to skip the check of the multicast packets source IP on the SBD IRB when PIM is configured in an SBD IRB. This requires the configuration of **protocols pim interface[=SBD] multicast-senders always**.

The following shows the configuration of EVPN multicast for Layer 3 (OISM):

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance SBD
    network-instance SBD {
        type mac-vrf
        interface irb0.0 {
        }
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
                    evi 1
                    supplementary-broadcast-domain {
                    }
                    routes {
                        route-table {
                            ip-prefix {
                                advertise-interface-ful true
                            }
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
            igmp-snooping {
                admin-state enable
                interface irb0.0 {
                }
            }
        }
    }
```

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance IP-VRF
    network-instance IP-VRF {
        type ip-vrf
```


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

```
        interface irb0.0 {
        }
        interface irb0.1 {
        }
        protocols {
            igmp {
                admin-state enable
                interface irb0.1 {
                }
            }
            pim {
                admin-state enable
                interface irb0.0 {
                }
                interface irb0.1 {
                }
            }
        }
    }
```

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance BD2
    network-instance BD2 {
        type mac-vrf
        interface irb0.1 {
        }
        interface lag-1.1 {
        }
        vxlan-interface vxlan0.2 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
                    evi 2
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
            igmp-snooping {
                admin-state enable
                interface irb0.1 {
                }
            }
        }
    }
```

```
--{ * candidate shared default }--[  ]--
# info with-context interface irb0
    interface irb0 {
        subinterface 0 {
            evpn-interface-ful-unnumbered {
            }
            ipv4 {
                admin-state enable
        subinterface 1 {
            ipv4 {
                admin-state enable
                address 101.1.1.254/24 {
                    anycast-gw true
```


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

```
                }
                arp {
                    learn-unsolicited true
                    evpn {
                        advertise dynamic {
                        }
                    }
                }
            }
        }
    }
```

OISM on 7250 IXR Gen 2c+

To enable OISM on 7250 IXR Gen 2c+ platforms, the following command is used: `network-instance[=ip-vrf] protocols bgp-evpn optimized-inter-subnet-multicast-forwarding`.

The following considerations apply to the **optimized-inter-subnet-multicast-forwarding** command:

- It is used to account for the IRBs in OISM mode and complete validation in the management server against the available datapath resources.
- An SBD IRB cannot be linked to an IP-VRF without this command enabled. If not configured, the configuration of an SBD in the IP-VRF fails upon commit.
- This command is only configurable in IP-VRFs and only when PIM and IGMP or MLD are enabled.
- This command is exclusive to 7250 IXR Gen 2c+ platforms and is not applicable to other platforms.

4.8.3.4 PEG function with ingress replication

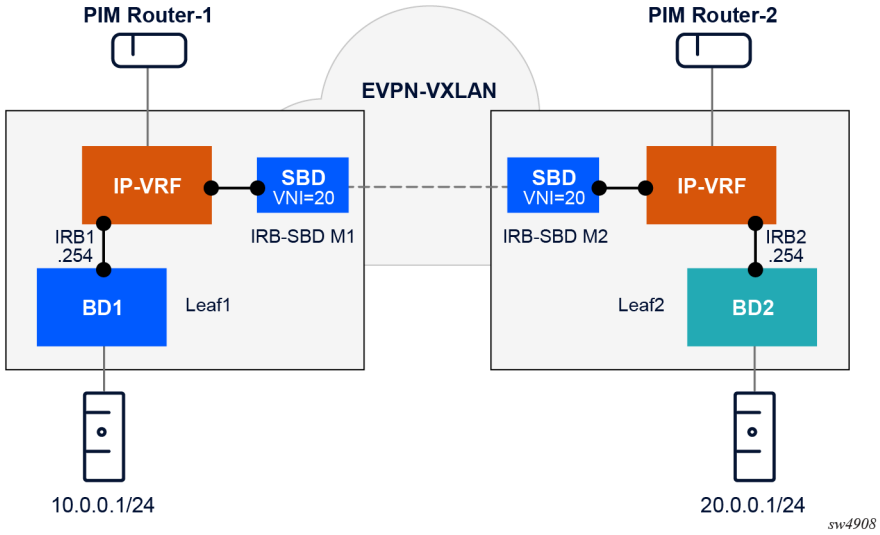
Note: This feature is supported on 7220 IXR-D3L, 7220 IXR-D2L, 7220 IXR-D5, 7220 IXR-D4, and 7250 IXR Gen 2c+ platforms.

EVPN OISM functions under the assumption that sources and receivers are always attached to broadcast domains (BD) via MAC-VRF interfaces. However, there are cases where some sources and receivers for a tenant are connected to an IP-VRF interface instead of a BD interface. In these cases, the IP-VRF Layer 3 multicast states may have IP-VRF non-BD interfaces or IP-VRF BD interfaces as IIF or OIF list members.

The following figure shows how the use of the PIM to EVPN gateway (PEG) function is applied to include PIM routers in EVPN OISM networks.

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

Figure 34: PEG function

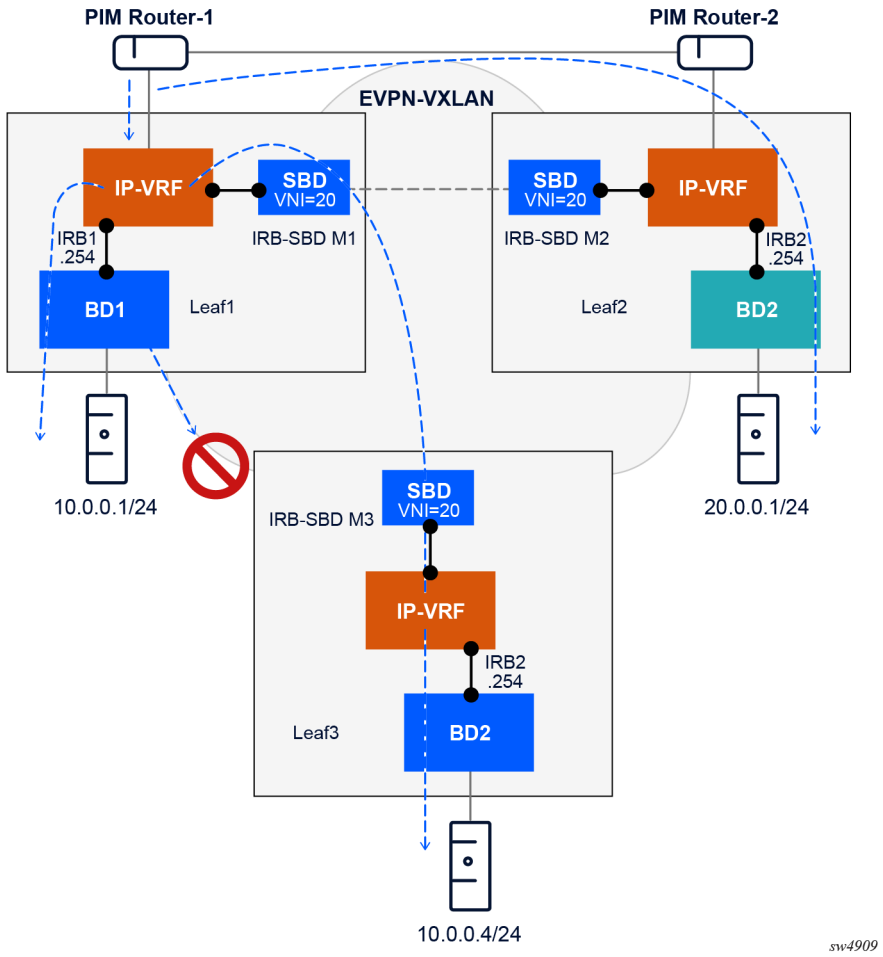


Data path requirements

The following figure illustrates the data path extensions for PEG functions.

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

Figure 35: PEG data path



A PE receiving multicast traffic on a local PIM-routed subinterface forwards multicast traffic directly to local receivers in BDs and to the supplementary broadcast domain (SBD) if it is the PEG Designated Router (DR). In the figure, Leaf1 sends multicast traffic to local BD1 interfaces and the SBD, but not to VXLAN binds on BD1.

A PE receiving multicast traffic from EVPN on the SBD forwards the traffic to local receivers through the SBD interface, and not to EVPN destinations. In the figure, Leaf3 receives multicast traffic on the SBD, forwards the traffic to BD2 bridge subinterfaces, but not to BD2 VXLAN binds.

Configuring PEG function

The following PEG configuration model comprises of an IP-VRF with a SBD and one or more ordinary BDs, with Layer 3 interfaces on the IP-VRF where PIM is configured.

The **evpn-multicast-gateway admin-state enable** command is configured only under the SBD. This command is used if the IP-VRF is configured with Layer 3 interfaces with PIM/IGMP/MLD enabled.

When **evpn-multicast-gateway** and `gateway-type peg` commands are configured in the SBD, the IMET route for the SBD is advertised with the SBD flag and the PEG flag.

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

The **dr-activation-timer** command has a default value of 3 seconds. This command is used in the DR election and is similar to the Ethernet segment activation timer for the ES DF election. After a PE is promoted to a DR, it advertises a wildcard SMET route.

The **non-dr-attract-traffic** command controls whether multicast traffic is attracted to the non-DR PEG from EVPN and from PIM/MVPN. This applies when fast convergence is required in the network.

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance SBD
    network-instance SBD {
        type mac-vrf
        interface irb0.0 {
        }
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
                    evi 1
                    supplementary-broadcast-domain {
                        evpn-multicast-gateway {
                            admin-state enable
                            gateway-type peg
                            non-dr-attract-traffic true
                            dr-election {
                                timers {
                                    dr-activation-timer 3
                                }
                            }
                        }
                    }
                    routes {
                        route-table {
                            ip-prefix {
                                advertise-interface-ful true
                            }
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
            igmp-snooping {
                admin-state enable
                interface irb0.0 {
                }
            }
        }
    }

--{ * candidate shared default }--[  ]--
# info with-context interface irb0
    interface irb0 {
        subinterface 0 {
            ipv4 {
                unnumbered {
                    admin-state enable
                }
            }
        }
```


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

```
    }

--{ * candidate shared default }--[  ]--
# info with-context network-instance IP-VRF
    network-instance IP-VRF {
        type ip-vrf
        interface ethernet-1/1.1 {
        }
        interface irb0.0 {
        }
        interface irb0.1 {
        }
        protocols {
            igmp {
                admin-state enable
                interface irb0.1 {
                }
            }
            pim {
                admin-state enable
                interface ethernet-1/1.1 {
                }
                interface irb0.0 {
                }
                interface irb0.1 {
                }
            }
        }
    }

--{ * candidate shared default }--[  ]--
# info with-context network-instance BD2
    network-instance BD2 {
        type mac-vrf
        interface irb0.1 {
        }
        interface lag-1.1 {
        }
        vxlan-interface vxlan0.2 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
                    evi 2
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                }
            }
            igmp-snooping {
                admin-state enable
                interface irb0.1 {
                interface lag-1.1 {
                }
            }
        }
    }
}

--{ * candidate shared default }--[  ]--
# info with-context interface irb0
    interface irb0 {
```


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

```
        subinterface 0 {
            ipv4 {
                unnumbered {
                    admin-state enable
                }
            }
        }
        subinterface 1 {
            ipv4 {
                address 101.1.1.254/24 {
                    anycast-gw true
                }
                arp {
                    learn-unsolicited true
                    evpn {
                        advertise dynamic {
                        }
                    }
                }
            }
            anycast-gw {
            }
        }
    }
```

4.9 BGP and routing policy extensions for EVPN

This chapter describes extensions added to SR Linux BGP and routing policy configuration to facilitate EVPN configuration.

4.9.1 BGP extensions for EVPN

SR Linux supports Multi-Protocol BGP with AFI/SAFI EVPN. The following BGP features are relevant to EVPN in a VXLAN network:

- The EVPN address family can use eBGP or iBGP.
- eBGP multihop is supported on SR Linux; local-AS override can be used for iBGP per session.
- A supported configuration or design with eBGP is eBGP with local-as override on the session to an iBGP RR.
- Rapid-update and rapid-withdrawal for EVPN family are supported and are always expected to be enabled, especially along with multihoming.

  Note: Rapid-update is address-family specific, while rapid-withdrawal is generic for all address families.

- The BGP keep-all-routes option is supported for EVPN to avoid route-refresh messages attracting all EVPN routes when a policy changes or BGP-EVPN is enabled.
- The **receive-ipv6-next-hops** option does not apply to the EVPN address family.
- The **prefix-limit max-received-routes** and **threshold** options are supported for EVPN.
- The command **network-instance protocols bgp route-advertisement wait-for-fib-install** does not apply to EVPN.

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

- SR Linux BGP resolves BGP-EVPN routes' next-hops in the network-instance default route-table. If the next-hop is resolved, BGP can mark the route as u*> where *u* is used, * is valid, and > is best) and send the route to evpn_mgr if needed or reflected to other peers.
- BGP dynamic neighbors for IPv4 and IPv6 families are fully supported in SR Linux. EVPN-VXLAN routes can be resolved to the IPv4 addresses advertised by the dynamic neighbors.
- The command **default-received-encapsulation <mpls|vxlan>**, added to the BGP configuration in the default network-instance, indicates the encapsulation considered when the EVPN routes are received without a BGP encapsulation EC.

  Most EVPN routes are usually received with a BGP encapsulation extended community (EC) that indicates the encapsulation and therefore how to interpret the value in the received Label fields of the NLRI. If no encapsulation is received, BGP validates the route as MPLS or VXLAN, depending on how this command is configured. The default is **vxlan**.

4.9.2 Best path selection for EVPN IFL routes

The SR Linux fib_mgr makes the route selection across owners for the same prefix in the network-instance route-table based on the following:

1. route-table PREFERENCE
2. METRIC
3. `source_id` (which is always zero for BGP-based routes)
4. `src_instance_id` (for leaked routes, the ID of the source network-instance)
5. `owner_id`, where `bgp` is preferred over `bgp-evpn`, and `bgp-evpn` is preferred over `bgp-ipvpn`.

For example, if the same prefix 10.0.0.0/16 is received on the same IP-VRF route table via BGP PE-CE (IPv4 family) and via EVPN-IFL, the route with lowest route table PREFERENCE wins. If both routes have the same PREFERENCE (for example, the default 170), the route with lowest METRIC wins (assuming `combined-ecmp` is not configured). The metric is a measurement of the distance to the next-hop, typically obtained from the IGP protocol that resolves the route.

As another example, if prefix 10.0.0.0/8 is received via BGP PE-CE and via EVPN IFL on the same IP-VRF, with same PREFERENCE and METRIC, the router typically selects the PE-CE route based on the lowest `owner_id`.

When an IP-VRF network-instance has multiple BGP paths to the same destination, the selection of the best path to install in the FIB is based on the criteria in "BGP best path selection" in the SR Linux Routing Protocols Guide.

4.9.3 Routing policy extensions for EVPN

SR Linux includes support for applying routing policies to EVPN routes. You can specify the following match conditions in a policy statement:

- match routes based on EVPN route type
- match routes based on IP addresses and prefixes via prefix-sets for route types 2 and 5
- match routes based BGP encapsulation extended community
- match routes based on configured internal-tags

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

For information about configuring routing policies on SR Linux, see the *SR Linux Configuration Basics Guide*.

The following considerations apply to routing policies for EVPN:

- For IBGP neighbors, EVPN routes are imported and exported without explicit configuration of any policy at either the BGP or network-instance level.
- For EBGP neighbors, by default, routes are imported or exported based on the **ebgp-default-policy** configuration.
- When an explicit import/export route-target is configured in a network-instance bgp-instance, and an import/export policy is also configured on the same bgp-instance, the configured policy is used, and its route-target is added to the imported/exported route.
- When a network-instance policy and a peer policy are applied, they are executed as follows:
  - For export, the network-instance export policy is applied first, and the peer policy is applied afterwards (sequentially).
  - For import, peer import policy is applied first and network-instance import policy is applied afterwards (sequentially).
- Only one network-instance export and import policy is supported.

4.9.4 Best path selection for EVPN AD per-EVI routes

Note: This feature is supported on 7220 IXR and 7250 IXR (excluding mixed chassis) platforms.

For EVPN AD per-EVI routes, BGP best path selection orders routes using selected BGP attributes, including when candidates have different route distinguishers (RDs).

When two or more EVPN routes are received at a PE, BGP route selection typically takes place when the route keys are equal. When the route keys are different (different RDs), BGP internally propagates the routes to EVPN and the EVPN application performs the default selection based on the following tie-breaking order:

1. Highest local preference
2. Shortest AS-PATH length (fewer ASN members in AS-PATH)
3. Lowest IP (next-hop IP of the EVPN NLRI)
4. Lowest Ethernet tag
5. Lowest RD
6. Lowest BGP instance (this tie-breaking rule applies only when the service has two BGP instances of the same encapsulation)

This ordering applies to EVPN AD per-EVI route handling across all service types that include MAC-VRF (Layer 2 aliasing), VPWS (Ethernet tag ID selection and Layer 2 aliasing), and IP-VRF (Layer 3 aliasing, including non-zero-ESI EVPN-IFL-HOST and non-zero-ESI EVPN-IFL routes).

The BGP best path selection for AD per-EVI routes includes a configurable ECMP break-off behavior with stricter ECMP candidate selection for AD per-EVI routes. Specifically, ECMP only includes routes tied on the winning priority attributes, instead of taking top-N valid routes regardless of attribute mismatch. This is enabled by the **ecmp-strict-path-selection** command.

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

Only routes tied on the best priority attributes (local preference and AS-PATH-length) are eligible for ECMP set construction. If fewer than the configured ECMP-width are tied at best attributes, only the smaller tied set is used.

ECMP break-off behavior:

- prevents including lower-priority AD per-EVI routes into the ECMP set when a smaller best-tied subset is available.
- enhances maintenance-related scenarios by keeping the forwarding candidates aligned to best-path criteria within each Ethernet segment ID's aliasing destination.
- aligns route-selection across EVPN service types.

```
--{ * candidate shared default }--[  ]--
# info with-context system network-instance protocols
    system {
        network-instance {
            protocols {
                evpn {
                    routes {
                        ad-per-evi {
                            ecmp-strict-path-selection {
                            }
                        }
                    }
                }
            }
        }
    }
```

When **ecmp-strict-path-selection** is disabled, lower-priority but valid AD per-EVI routes can appear in the ECMP set (up to the ECMP width).

When **ecmp-strict-path-selection** is enabled, only best-tied routes are included.

When one route has a higher local preference, that route must be selected over alternatives even if those routes have a shorter AS-PATH.

4.10 Route internal tags (tag-sets in routing policies)

Route tags are pieces of information or identifiers attached to individual routes. They can be either external (protocol) tags or internal tags:

- External (protocol) tags are sent on the wire and are typically part of a routing protocol; for example, IS-IS route tags and OSPF route tags.
- Internal tags are not sent on the wire, and they have meaning only within an individual system. Internal tags are assigned to routes by policies or specific commands, and are used by policies to match on a group of routes. Internal tags are typically used to facilitate the matching of multiple routes at the same time or propagate route attributes between different routing protocols.

This chapter describes SR Linux support for internal tags and how to configure them.

About internal tags

In SR Linux, route internal tags are data structures that are attached to IP routes in the FIB or EVPN-specific routes. Internal tags are useful for carrying protocol information such as communities, area-id, as-

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

path, and so on, from one routing protocol context to another. For example, BGP can push its attribute fields, such as local preference, into the FIB as an internal tag. On export, the internal tag associated with the route can be matched by export policies to map the tag into any other metric for a routing protocol.

Every FIB route can be associated with an array of tags. The number of array elements is limited to a maximum of 2 in the current release.

To define internal tags in SR Linux, you configure tag-sets at the `routing-policy` level, and refer to the tag-sets by name at other configuration levels. See Configuring route internal tags.

SR Linux applications that support internal tags

The following SR Linux applications can write internal tags:

- EVPN and Layer 2 proxy-ARP/ND
  EVPN and Layer 2 proxy-ARP/ND support writing internal tags for all MAC/IP advertisement routes generated out of the active MACs in the mac-table or IP-to-MAC bindings in the proxy-ARP/ND tables, respectively.
- Layer 3 ARP/ND
  SR Linux supports writing internal tags into ARP/ND entries that generate an EVPN MAC/IP advertisement route and also into the ARP/ND entries that generate an arp-nd host route into the route-table.
- Layer 3 routes
  In SR Linux, BGP can write internal tags into the route-table entries for a routed network-instance. In this case, all routes to be exported as prefix routes are assigned the configured internal tag.

  The route-table includes a state leaf-list with the allocated `internal-tags` tag values that indicate whether a specific route entry is associated with an internal tag.

Routing policies and internal tags

An import or export policy can write an internal tag as a configured `action` or `default-action`. These actions are supported on BGP policies applied to the supported families; that is, `ipv4-unicast`, `ipv6-unicast`, and `evpn`.

When internal tags are used in BGP import policies, the policies can be applied at the default network-instance level (including global, per-group, or per-peer policies), MAC-VRF network-instance level, or IP-VRF network-instance level.

When internal tags are used in BGP export policies, the policies can be applied at the MAC-VRF or IP-VRF network-instance level, or IP-VRF/default network-instance level for inter-instance policies.

When an import policy matches an IPv4 BGP route, IPv6 BGP route, or EVPN IP prefix route (route type 5) and adds a tag-set, the corresponding route created in the route-table contains the internal tag and is exposed in the route-table entry `internal-tags`.

When the import policy matches an EVPN MAC/IP advertisement route (route type 2) and adds a tag-set as an action, the corresponding created MAC and/or ARP/ND entry is associated with the internal tag. Matching on any other EVPN route type and adding an internal tag has no effect, since the rest of the EVPN route types do not create entries into the MAC table, ARP/ND table, or route table.

A MAC/IP VRF export policy can add a tag-set as an action in the following cases

- On a MAC-VRF, the `bgp-vpn.bgp-instance.export-policy` can match EVPN MAC/IP routes (with or without IP address) generated for dynamic or static entries in the mac-table or proxy-ARP/ND table; that is, basically for any entry that generates a MAC/IP advertisement route in the context of the

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

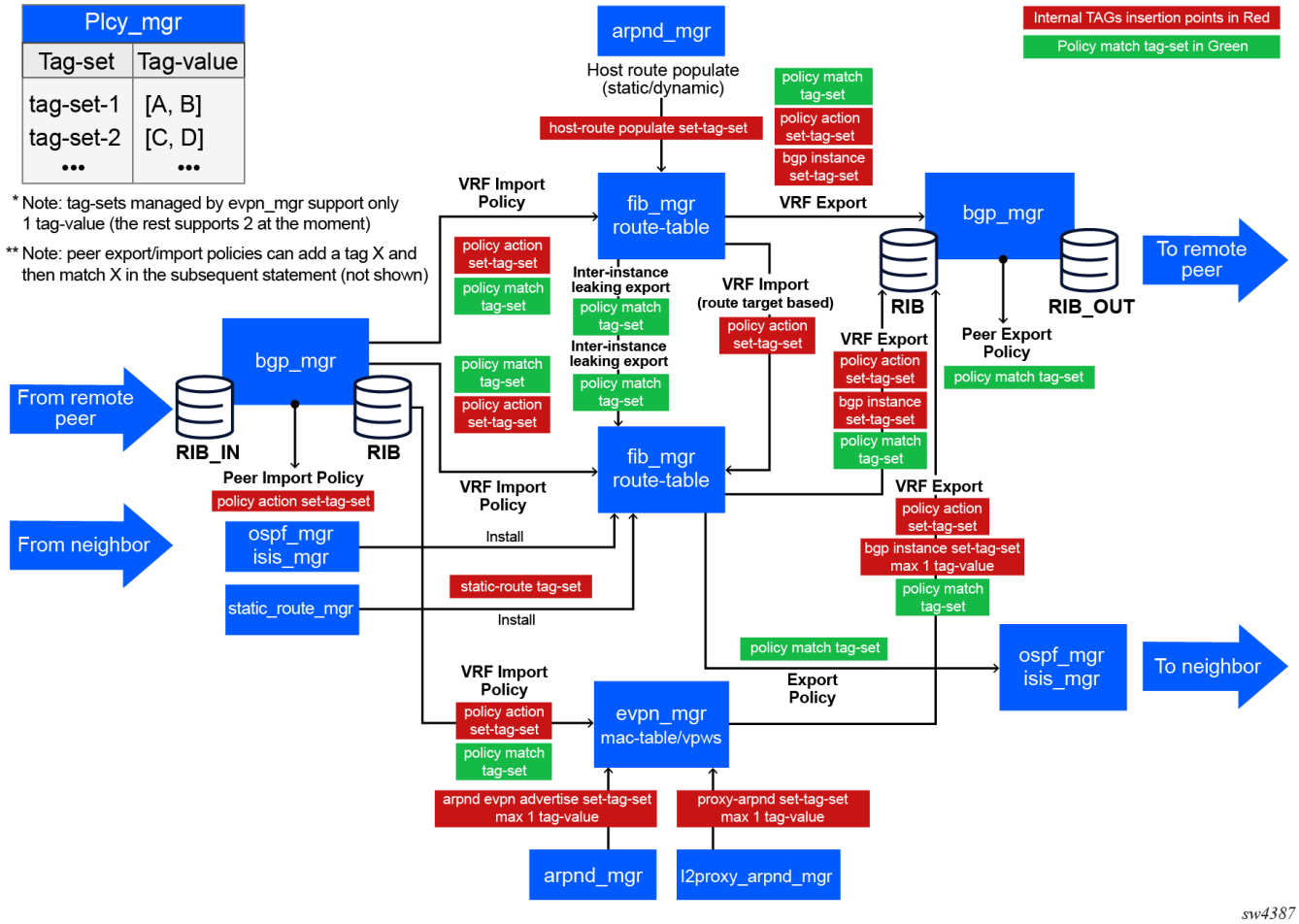
MAC-VRF. In those policies, a tag-set can be added. The same tag-set can be matched on peer/group-level export policies in the default network-instance.

- On an IP-VRF, the `bgp-vpn.bgp-instance.export-policy` can match any route-table entry that generates an EVPN IP prefix route in the context of the IP-VRF. In these policies, a tag-set can be added as an action, and the same tag-set can be matched by a peer/group-level export policy in the default network-instance.
- On an IP-VRF or default network-instance, the `inter-instance-policies.export-policy` can match any route-table entry and make those entries available for leaking. The internal tag can be used by an `inter-instance-policies.import-policy` to match the routes that are actually leaked into the destination Layer 3 network-instance.

Insertion points for internal tags

The following diagram shows the SR Linux applications that can write an internal tag, insertion points for the internal tags, and the policy points where internal tag matching can take place.

Figure 36: Applications for internal tags, insertion points, and policy points



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

4.10.1 Order of precedence when multiple applications write internal tags

As described in the previous section, multiple SR Linux applications can write internal tags for a route. For example, a newly created ARP-ND host 10.0.0.1/32 route can be associated with tag-set `TAG-1`, and at the same time an export policy applied on IP-VRF-1, where 10.0.0.1/32 is installed, may have an action to write tag-set `TAG-2` or the same route.

In this case, SR Linux applies precedence rules to determine the tag-set that gets applied to the route. The following sections describe the precedence rules applied when multiple applications attempt to write an internal tag on the same route.

Writing a tag for MAC/IP routes generated for a MAC-VRF

The order in which the internal tag is applied is as follows:

(proxy-arp/nd tag (mac-vrf) + vrf-export-policy tag (mac-vrf level)) > (bgp-evpn.bgp-instance tag (mac-vrf) +vrf-export-policy tag (mac-vrf level))

That is, up to two tag-values can be written. However, if the route already has a tag-value, then the bgp-evpn.bgp-instance tag is not used. The vrf-export-policy tag action always appends to the route's tag (if already present on the route), or the default tag (under the bgp-instance) if configured.

Writing a tag for MAC/IP routes generated for a MAC-VRF with IRB

(arp/nd evpn tag (irb) + vrf-export-policy tag (mac-vrf level)) > (bgp-evpn.bgp-instance tag (mac-vrf) +vrf-export-policy tag (mac-vrf level))

In the same way as proxy-arp/nd tags, up to two tag-values can be written, and if the route already has a tag-value, then the bgp-evpn.bgp-instance tag is not used. The vrf-export-policy tag action always appends to the route's tag (if already present on the route), or the default tag (under the bgp-instance) if configured.

Writing a tag for ARP-ND host routes in an IP-VRF

Tags are appended until a maximum of two is reached, so the vrf-export policy action does not overwrite the tag if that is already present. The default tag of the instance is used when arpnd does not add a tag.

A peer-export policy uses this precedence:

(arp/nd host-route tag (irb) + vrf-export policy tag) > (bgp-evpn.bgp-instance tag (ip-vrf) + vrf-export policy tag)

Writing a tag for IP routes in an IP-VRF (irrespective of the owner)

Tag-values are appended until a maximum of two is reached, so the vrf-export policy action does not overwrite the import-policy tag if that is already present. The default tag of the instance is used when import-policy does not add a tag.

A peer-export policy uses this precedence:

(peer-import-policy-tag + vrf-export policy tag) > (bgp-evpn.bgp-instance tag (ip-vrf) + vrf-export policy tag)

In cases where the routes are static routes, IS-IS routes, OSPF routes, and so on, when exported into EVPN IP prefix routes, there is no peer-import-policy.

In cases where there are BGP PE-CE routes that are imported into the IP-VRF route-table and readvertised as EVPN IP prefix routes, there are peer-import-policies, and the preference is as above.

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

Writing a tag for MAC/IP routes in MAC-VRF or routes in IP-VRF via multiple import policies

If multiple import-policies match the same route and try to write a tag, writing is allowed until a maximum of two tag-values is reached. Note that applying "multiple import-policies" refers to policy chaining at the peer-import level.

Writing a tag for routes via vrf-import and peer import policies

If both vrf-import and peer-import policies apply to a route, and each step inserts a tag-set, then the tag-set applied as a result of the vrf-import policy takes precedence.

Writing a tag for BGP routes in an ECMP set

If BGP creates an ECMP set for a prefix, when installing such a prefix in the FIB, BGP copies the tag-set of the best route into fib_mgr. This is what other protocols would see when importing the prefix from fib_mgr.

This also applies to BGP VPN routes that are received with different RDs and combined in an ECMP set.

4.10.2 Configuring route internal tags

Procedure

To configure route internal tags, you configure a tag-set and configure one or more applications to write the tag-set to routes.

Example: Configure a tag-set

```
--{ * candidate shared default }--[  ]--
# info with-context routing-policy
    routing-policy {
        tag-set ts1 {
            tag-value [
                1
                2
            ]
        }
    }
```

Example: Match a tag-set in a routing policy

```
--{ +* candidate shared default }--[  ]--
# info with-context routing-policy policy match_tag_1
    routing-policy {
        policy match_tag_1 {
            statement 1 {
                match {
                    internal-tags {
                        tag-set [
                            ts1
                        ]
                    }
                }
            }
        }
    }
```


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

Example: Write a tag-set as an action in a routing policy

```
--{ * candidate shared default }--[  ]--
# info with-context routing-policy policy d1
    routing-policy {
        policy t1 {
            default-action {
                internal-tags {
                    set-tag-set [
                        ts1
                    ]
                }
            }
        }
    }
```

Example: Write internal tags for MAC/IP advertisement routes generated out of the active MACs in the MAC table

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance vrf1
    network-instance vrf1 {
        type mac-vrf
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    internal-tags {
                        set-tag-set [
                            ts1
                        ]
                    }
                }
            }
        }
    }
```

Example: Write internal tags for IP-to-MAC bindings in proxy-ARP/ND tables

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance vrf1 bridge-table proxy-arp evpn
    network-instance vrf1 {
        bridge-table {
            proxy-arp {
                evpn {
                    internal-tags {
                        set-tag-set [
                            ts1
                        ]
                    }
                }
            }
        }
    }
```

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance vrf1 bridge-table proxy-nd evpn
    network-instance vrf1 {
        bridge-table {
            proxy-nd {
                evpn {
```


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

```
                    internal-tags {
                        set-tag-set [
                            ts1
                        ]
                    }
                }
            }
        }
    }
```

Example: Write internal tags into ARP/ND entries that generate an EVPN MAC/IP advertisement route

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 subinterface 1 ipv4 arp evpn
    interface ethernet-1/1 {
        subinterface 1 {
            ipv4 {
                arp {
                    evpn {
                        advertise dynamic {
                            internal-tags {
                                set-tag-set [
                                    ts1
                                ]
                            }
                        }
                    }
                }
            }
        }
    }
```

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 subinterface 1 ipv6 neighbor-discovery
    interface ethernet-1/1 {
        subinterface 1 {
            ipv6 {
                neighbor-discovery {
                    evpn {
                        advertise dynamic {
                            internal-tags {
                                set-tag-set [
                                    ts1
                                ]
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Write internal tags into ARP/ND entries that generate an ARP/ND host route into the route table

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 subinterface 1 ipv4 arp
    interface ethernet-1/1 {
        subinterface 1 {
            ipv4 {
```


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

```
                arp {
                    host-route {
                        populate evpn {
                            internal-tags {
                                set-tag-set [
                                    ts1
                                ]
                            }
                        }
                    }
                }
            }
        }
    }
```

```
--{ * candidate shared default }--[  ]--
# info with-context interface ethernet-1/1 subinterface 1 ipv6 neighbor-discovery
    interface ethernet-1/1 {
        subinterface 1 {
            ipv6 {
                neighbor-discovery {
                    host-route {
                        populate evpn {
                            internal-tags {
                                set-tag-set [
                                    ts1
                                ]
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Write internal tag into routes to be exported as EVPN IP prefix routes

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance ip-vrf-1 protocols bgp-evpn bgp-instance 1
    network-instance ip-vrf-1 {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    internal-tags {
                        set-tag-set [
                            ts1
                        ]
                    }
                }
            }
        }
    }
```

4.11 EVPN VXLAN Anycast Multi-Homing

This chapter describes the implementation of Anycast Multi-Homing (MH).

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

4.11.1 EVPN VXLAN Anycast Multi-Homing for Layer 2 services

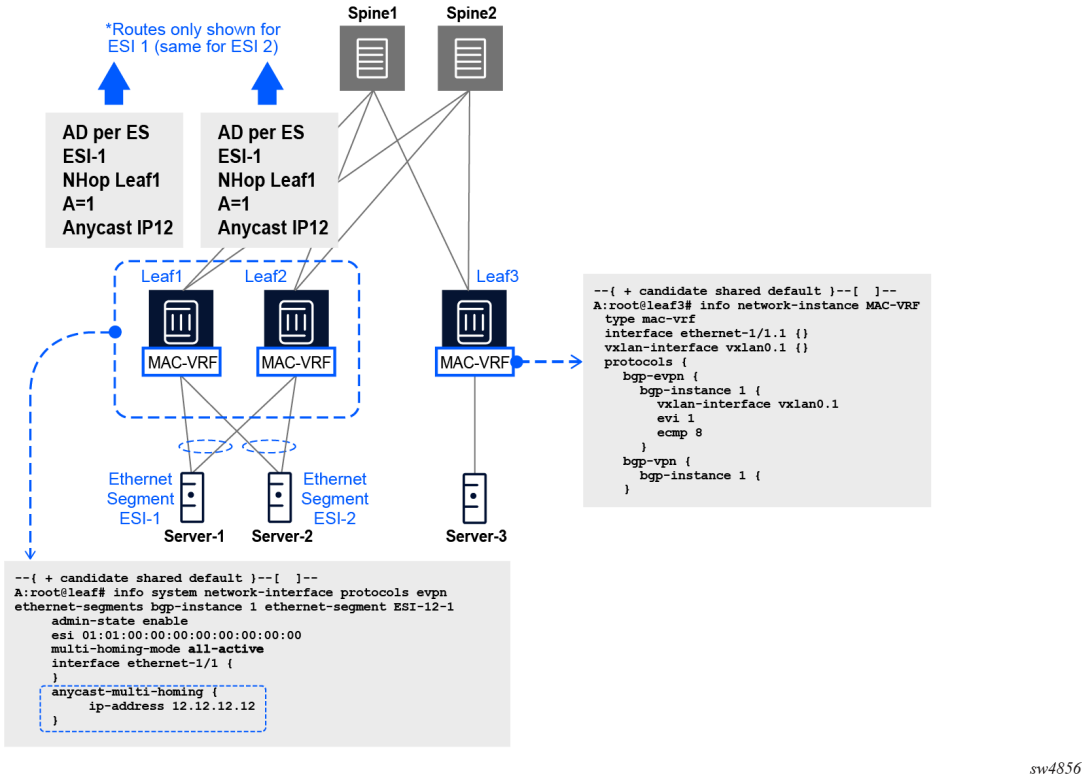
This chapter describes the implementation of Anycast Multi-Homing (MH) for Layer 2 services.

4.11.1.1 Anycast MH

EVPN VXLAN Anycast Multi-Homing (MH) leverages Anycast VXLAN Termination Endpoints (VTEPs) shared by the multi-homed leaf routers as defined in *draft-rabnag-bess-evpn-anycast-aliasing* (Anycast Multi-Homing for EVPN).

The following figure shows a leaf–spine fabric where Leaf1 and Leaf2 act as egress leaf nodes that multihome Server-1 and Server-2, while Leaf3 node serves as the ingress leaf forwarding traffic toward them. This figure serves as a reference point for all subsequent sections throughout this chapter.

Figure 37: EVPN Anycast MH



sw4856

- The egress leaf is the system attached to the Ethernet Segment (ES) working in the Anycast MH mode.
- The ingress leaf is the system that performs the Ethernet Aliasing function (Overlay ECMP).

Supported platforms

Anycast Multi-Homing is supported on the following platforms:

- 7220 IXR-D*x* (ingress and egress function)
- 7250 IXR Gen 2c+ (ingress and egress leaf function)

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

- 7250 IXR Gen 3 (ingress and egress leaf function)
- 7220 IXR-H5 (ingress and egress function)

4.11.1.1.1 Configuring Anycast MH

Procedure

Anycast Multi-Homing (MH) is enabled per Ethernet Segment (ES) on egress leaf nodes (such as Leaf1 and Leaf2 in Figure 37: EVPN Anycast MH).

Note:
The ingress leaf node (such as Leaf3 in Figure 37: EVPN Anycast MH) requires no specific configuration to enable the Anycast MH mode. For backward compatibility within the same MAC-VRF, the ingress leaf supports two types of overlay ECMP.

- ECMP to all egress leaf nodes advertising the Anycast VTEP
- Ethernet Aliasing to all nodes of the same ES that advertise A-D per EVI routes and are operating in a non-anycast mode.

To configure Anycast MH per ES on the egress leaf node, use the command **ethernet-segment.anycast-multi-homing**. The configuration of **ethernet-segment.anycast-multi-homing.ip-address <ip-address>** is mandatory.

The IP address parameter (**ip-address <ip-address>**) should match an IPv4 address configured on a loopback interface in the default network instance. The system does not verify that the address exists on a loopback interface; you must ensure the correct IP address is configured.

The Anycast MH mode is supported in the following scenarios:

- All-active multihoming ES
- ES associated with Ethernet or LAG interfaces

Example

The following configuration example enables Anycast MH for the ES named `srl_ES` within BGP instance 1, assigning 12.12.12.12 as the shared Anycast IP address for that segment.

```
--{ * candidate shared default }--[  ]--
# info with-context system network-instance protocols evpn ethernet-segments
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment srl_ES {
                                anycast-multi-homing {
                                    ip-address 12.12.12.12
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```


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

4.11.1.2 Egress leaf in Anycast MH mode

The following sections describe the egress leaf working in Anycast Multi-Homing (MH) mode (such as Leaf1 and Leaf2 in Figure 37: EVPN Anycast MH).

4.11.1.2.1 Prerequisites

Following are the prerequisites for egress leaf nodes (such as Leaf1 and Leaf2 in Figure 37: EVPN Anycast MH) to operate in Anycast Multi-Homing (MH) mode:

- Each egress leaf node must be configured with the same loopback IP address (**ethernet-segment.anycast-multi-homing.ip-address <ip-address>**). This IP address is referred to as the Anycast VTEP. This Anycast VTEP must be advertised in the underlay routing protocol (IS-IS, OSPF, or BGP) to ensure reachability from ingress leaf and spine routers.

  Note: The Anycast VTEP's loopback state is independent of the Ethernet Segment (ES) state. If the loopback goes down, the A-D per ES route is still advertised.

- All egress leaf routers attached to the same ES in Anycast MH mode must use the same Anycast VTEP address.

  Note: If the user wants to bring down the loopback interface when an ES goes down, Event Handling System (EHS) can be used.

- All egress leaf routers connected to the same ES in Anycast MH mode must use the same ingress VNI for each MAC-VRF.

  Note: Asymmetric VNIs are not supported for egress leaf nodes in Anycast MH mode.

- The Anycast VTEP feature requires VXLAN termination on at least two distinct IP addresses: the system IP and a non-system IPv4 address. Egress leaf nodes must support VXLAN termination across multiple loopback interfaces.

4.11.1.2.2 Egress leaf behavior in Anycast MH mode

When an egress leaf (such as Leaf1 and Leaf2 in Figure 37: EVPN Anycast MH) operates in Anycast MH mode for a specific ES, the following behaviors apply:

- Designated Forwarder (DF) Election and Local Bias behavior are unchanged. However, because A-D per-EVI routes are absent in this mode, the **ac-df exclude** capability must be configured, which then uses the **preference** algorithm.
- The advertisement of the Ethernet Segment Identifier (ESI) with EVPN routes remains unchanged.
- The processing and advertisement of BUM (Broadcast, Unknown Unicast, and Multicast) bindings and Inclusive Multicast Ethernet Tag (IMET) routes are not changed. The forwarding procedures for BUM traffic remain unchanged.
- Only the procedures related to the forwarding of unicast traffic to a remote ES are updated.

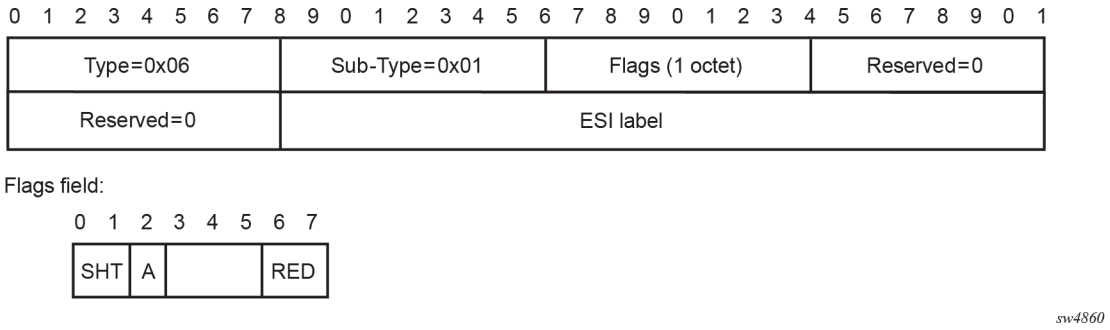
© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

4.11.1.2.3 Control plane behavior

From a control-plane perspective, an egress leaf working in Anycast MH mode (such as Leaf1 and Leaf2 in Figure 37: EVPN Anycast MH) for a given ES works as follows:

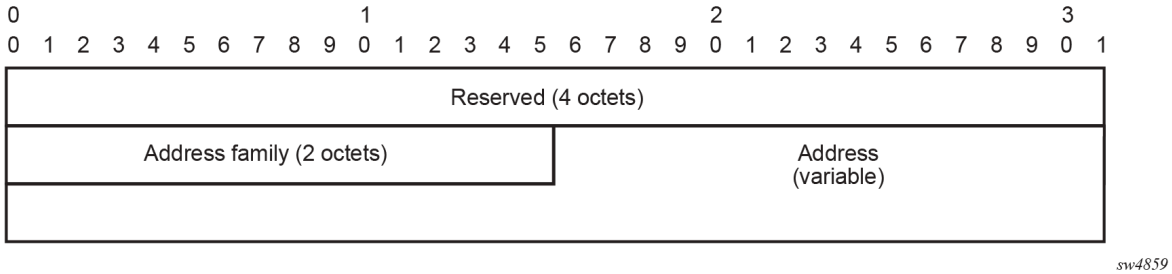
- AD per-EVI routes for Anycast MH ESes are suppressed based on the Anycast MH mode configuration.
  - AD per-EVI routes are suppressed only for MAC-VRFs with VXLAN encapsulation.
  - This applies to nodes sharing ESs by **mac-vrfs** with **vxlan** and **mpls encap**.
- AD per-ES routes for Anycast MH ESes are advertised with the **A=1 flag** in ESI label extended community (as defined in draft-rabnag-bess-evpn-anycast-aliasing).
  - The **anycast-multi-homing** flag indicates to remote NVEs that EVPN MAC/IP Advertisement routes with matching Ethernet Segment Identifier are resolved only by A-D per ES routes for the ES. This flag signals to the ingress leaf that no A-D per EVI routes are advertised for the Ethernet Segment. The A flag is bit 2 in the Flags field of the ESI label extended community.

    Figure 38: ESI label extended community



  - The BGP tunnel-encapsulation attribute (per draft-rabnag-bess-evpn-anycast-aliasing) carries the Anycast VTEP; specifically, it is encoded in the BGP Tunnel Encapsulation Attribute Tunnel Egress Endpoint Sub-TLV (code point 6) [RFC9012]. The Tunnel Egress Endpoint Sub-TLV Value Field is shown below.

    Figure 39: Tunnel egress endpoint Sub-TLV



- MAC/IP routes for MACs learned on the ES working in Anycast MH mode are advertised with ESI. The next-hop is either the regular `system0.0` IPv4 address or the one determined by **network-instance[=mac-vrf].protocols.bgp-evpn.bgp-instance.routes.bridge-table.next-hop <ip-address>**

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

4.11.1.2.4 Data plane behavior

From a data-plane perspective, an egress leaf working in Anycast MH mode (such as Leaf1 and Leaf 2 in Figure 37: EVPN Anycast MH) for a given ES works as follows:

- All VXLAN traffic is sourced from system0.0.
- The Anycast VTEP is never used as a source VTEP.

4.11.1.3 Ingress leaf in Anycast MH mode

The ingress leaf node (such as Leaf3 in Figure 37: EVPN Anycast MH) function is mainly a control-plane function and is supported on all platforms, without requiring any datapath extensions.

For creating the Ethernet Segment (ES) destination, the following four operation modes are supported.

Non-Anycast mode

The ingress leaf operates in this mode when the ESI is identified as a non-Anycast MH ES. A non-Anycast MH ES is an ES where all AD per ES routes are received with A=0.

Anycast mode with multiple AD per-ES routes

In this mode, the ingress leaf receives multiple AD per-ES routes for the same ESI, all with the 'A' flag set to 1 and a consistent Anycast VTEP.

The ES destination is created using the Anycast VTEP learned from these routes and the VNI from the first received MAC/IP Advertisement or IP Prefix route.

Anycast mode with a single AD per-ES route

This mode applies when only a single AD per-ES route is received for the ES, or when all but one of the AD per-ES routes have been withdrawn.

The ES destination is created using the next-hop from the AD per-ES route and the VNI from the first received MAC/IP Advertisement or IP Prefix route.

Mismatch mode

This mode occurs when multiple AD per-ES routes are received for the same ESI, but the 'A' flag or Anycast VTEP information is inconsistent. A single route with A=1 is enough for the ingress leaf to be in "anycast mode".

If the ingress leaf receives AD per ES routes with an inconsistent "A" flag or Anycast VTEP information, the ES destination is created using the VTEPs of the AD per-ES routes NLRI next hops (allows load balancing) and the VNI from the first received MAC/IP Advertisement or IP Prefix route.

Only resolved AD per-ES routes (those with a reachable next hop and Anycast VTEP) are considered.

If a route cannot be resolved, BGP applies treat-as-withdraw behavior.

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

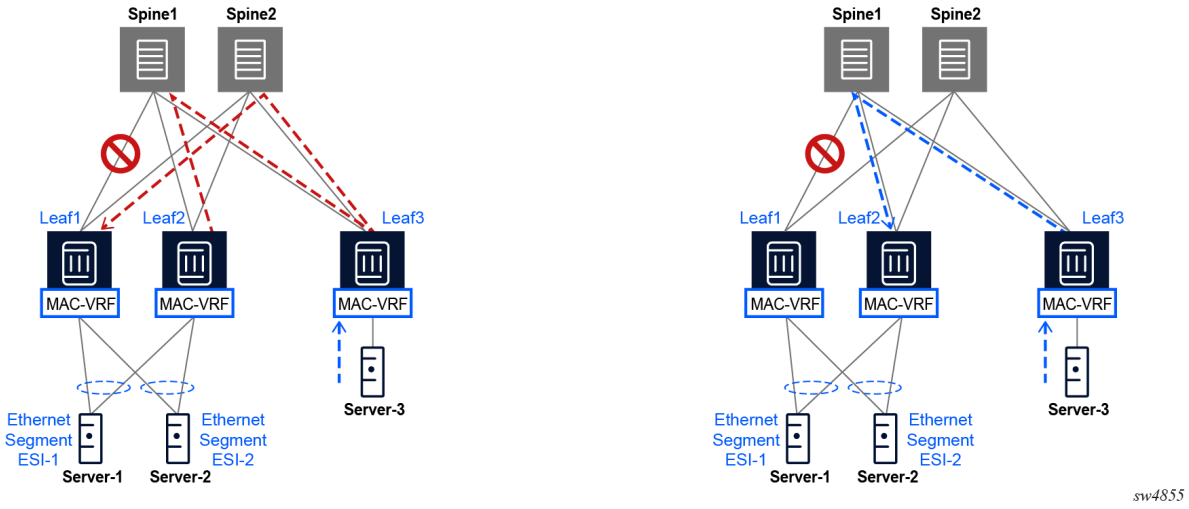
4.11.1.3.1 Ingress leaf BGP resolution of the Anycast VTEP

For EVPN routes advertised with an Anycast VTEP, successful resolution requires both the Network Layer Reachability Information (NLRI) Next Hop and the Anycast VTEP to be reachable. If either of these addresses becomes unreachable, the routes are not installed.

4.11.1.4 Comparing underlay failure behavior

The diagram below shows network behavior under default aliasing (left) versus Anycast aliasing (right).

Figure 40: Underlay failure: default aliasing versus Anycast aliasing



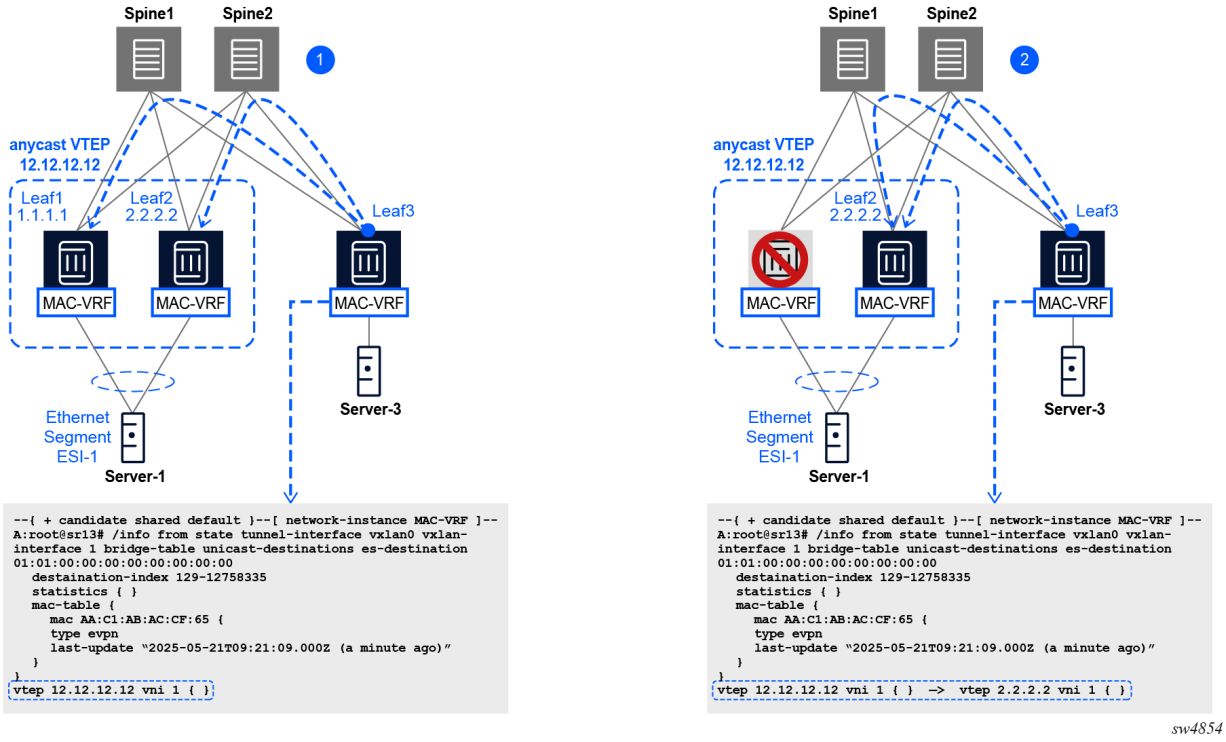
- Default aliasing mode: When an underlay network failure occurs, in-flight packets may path hunt until the underlay network converges. If the underlay uses eBGP and the spines share the same ASN, in-flight packets are dropped.
- Anycast aliasing mode: When an underlay network failure occurs, in-flight packets go directly from Spine1 to Leaf2. Therefore, convergence is faster and more efficient than in the default aliasing mode.

4.11.1.5 Egress leaf node failure behavior in Anycast ES

Anycast VTEPs ensure node failure events are handled smoothly and efficiently. The following diagram shows the situation before the node failure (left) and after (right).

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

Figure 41: Egress leaf node failure behavior in Anycast ESes



Note:
The following behavior assumes that Leaf3 keeps at least one MAC/IP route for Server-1 and ESI-1 after the node failure. Otherwise, the MAC is flushed, and unknown unicast traffic is flooded.

- Non-failure state: The ES destination on Leaf3 points to the Anycast VTEP IP address 12.12.12.12. Per flow load balancing occurs based on the underlay set to 12.12.12.12 without using overlay ECPM sets.
- Egress leaf failure (such as when Leaf1 goes down): In-flight packets are immediately rerouted to Leaf2, which also advertises the Anycast VTEP address. When Leaf3 receives the A-D per-ES MP_NLRI_UNREACH message from Leaf1, Leaf3 reprograms the ES destination to use the unicast VTEP of Leaf2, as specified by the AD per-ES route of the surviving ES leaf, Leaf2.

4.11.2 EVPN VXLAN Anycast Multi-Homing for Layer 3 services

Note: The components of EVPN VXLAN Anycast Multi-Homing for Layer 3 services are supported on 7250 IXR Gen 2c+, 7250 IXR Gen 3, 7220 IXR-D*x*, 7220 IXR-H5, and 7220 IXR-H6 platforms.

This chapter describes the implementation of Anycast Multi-Homing (MH) for Layer 3 services.

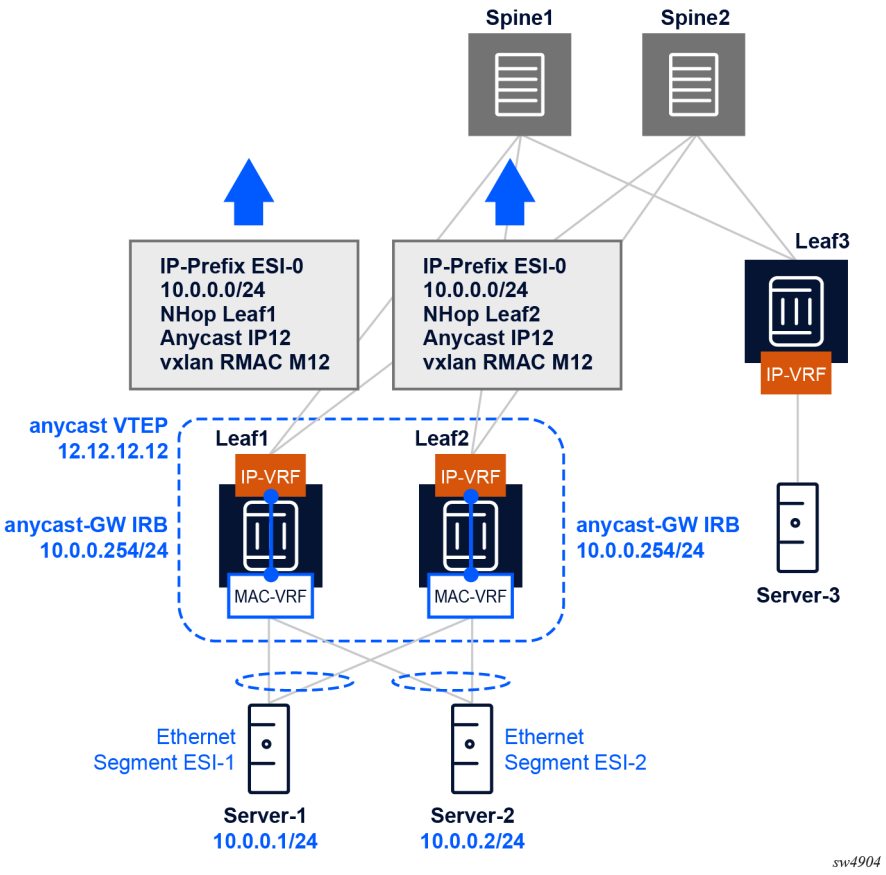
© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

4.11.2.1 EVPN VXLAN anycast aliasing for Layer 3 routes with ESI-0

Anycast MH can be applied to Ethernet Segment Identifier (ESI)-0 EVPN IFL routes advertised from two or more egress leaf routers.

The following figure illustrates anycast MH for ESI-0 EVPN IFL routes.

Figure 42: Anycast MH for ESI-0 EVPN IFL routes



Configuring anycast MH for EVPN IFL subnets

The following configuration enables the use of anycast MH for EVPN IFL subnets.

```
--{ * candidate shared default }--[  ]--
# info with-context network-instance ip-vrf
    network-instance ip-vrf {
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    routes {
                        route-table {
                            ip-prefix {
                                anycast-multi-homing {
                                    ip-address 12.12.12.12
                                }
```


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

```
                        }
                    }
                }
            }
        }
    }
}
```

Egress leaf in anycast MH mode for ESI-0

Two or more egress leafs that are attached to the same IP subnet and operate in anycast MH mode must use the same VNI to identify the IP-VRF associated with that IP subnet.

A single anycast VTEP is shared by the group of NVEs associated with the same IP subnets. These NVEs advertise reachability for the anycast VTEP in the underlay routing protocol. ESI-0 EVPN IFL routes are advertised with the anycast VTEP encoded in a BGP tunnel encapsulation attribute.

The presence of an anycast VTEP in the IP prefix route indicates to the ingress NVE that the prefix is using anycast MH. ESI-0 EVPN IFL routes do not carry the ESI extended community label; therefore they do not carry the A flag. For more information, see Anycast MH.

The router MAC used for anycast MH in RT5s (and IP A-D per ES routes in non-zero ESI Layer 3 cases; see EVPN VXLAN anycast aliasing for Layer 3 routes with non-zero ESI) is configured using the **system network-instance protocols vxlan router-mac-address <mac>** command and overrides the system MAC used in routed VXLAN interfaces as inner **source-mac** and **router-mac** in the EVPN routes. Only one **router-mac** per system can be configured and the **router-mac** is exposed in the state for routed VXLAN interfaces in the **inner-ethernet-header used-source-mac <mac>** context.

Ingress leaf in anycast MH mode for ESI-0

If GW-IP and anycast VTEP are both present in an RT5 route, BGP uses GW-IP for path selection and ignores anycast VTEP received in the RT5 route. If non-zero ESI and anycast VTEP are both present in an RT5 route, BGP prefers ESI and ignores anycast VTEP.

When receiving and importing ESI-0 EVPN IFL routes with an anycast VTEP for the same prefix, the following rules apply:

- When all contributing RIB-INs are consistent, the anycast VTEP is used for installing the route.
- For a specific prefix, consistency requires that all RIB-INs advertise the same anycast VTEP address (BGP Tunnel Encapsulation Attribute) and the same EVPN router's MAC extended community (GW-MAC). A differing VNI across RIB-INs does not affect consistency.
- If the anycast VTEP address differs across RIB-INs or the router's MAC differs across RIB-INs, the ingress leaf enters mismatch mode and falls back to installing the route using the BGP NLRI next hops of the contributing routes.
- If some RIB-INs carry an anycast VTEP and others do not (no Tunnel Encapsulation Attribute), the ingress leaf programs each contributing path individually. Routes without an anycast VTEP use the BGP NLRI next hop, and routes with an anycast VTEP use the anycast VTEP (ECMP across the resulting next-hop set).

When using anycast forwarding, the ingress leaf encapsulates the packets destined for a multihomed IP subnet into a VXLAN packet with the destination VTEP set to the anycast VTEP and the VNI set to the VNI of the best IP prefix route. All egress leafs attached to the multi-homed IP subnet have previously advertised reachability to the anycast VTEP, so the ingress leaf creates an underlay ECMP set for the anycast VTEP (assuming multiple equal-cost underlay paths).

The ingress leaf does not use an anycast VTEP as the outer source IP address of the VXLAN tunnel.

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

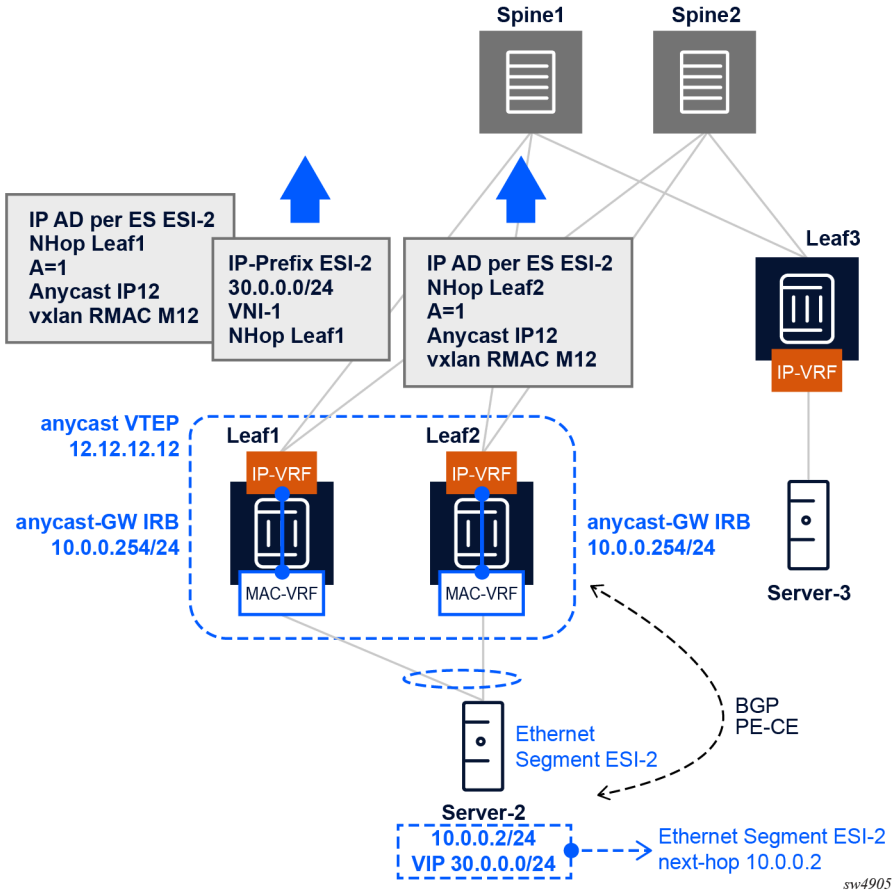
The anycast MH solution for Layer 3 does not support BGP fast reroute (FRR) `backup-paths`. FRR is not blocked in IP-VRFs that use anycast MH, but FRR only applies to RT5 routes without the anycast VTEP attribute.

ESI-0 EVPN routes with non-zero GW-IPs resolved to anycast MH EVPN-IFL-HOST routes are supported.

4.11.2.2 EVPN VXLAN anycast aliasing for Layer 3 routes with non-zero ESI

The following figure illustrates IP aliasing for non-zero Ethernet Segment Identifier (ESI) EVPN IFL routes.

Figure 43: Anycast MH for non-zero ESI EVPN IFL routes



Support for IP aliasing for non-zero ESI EVPN IFL routes is the same as for EVPN-IFL-HOST routes; for more information, see Multi-homed non-zero ESI EVPN-IFL-HOST routes

Anycast Ethernet Segments in the egress leaf routers are associated with a Layer 3 next hop (`ethernet-segment <string> next-hop <l3-next-hop>`) instead of an Ethernet or LAG interface.

Egress leaf in anycast MH mode for non-zero ESI

Anycast MH mode for non-zero ESI EVPN-IFL IP prefix routes is configured using the **ethernet-segment anycast-multi-homing** command.

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

The egress leaf in anycast MH mode for non-zero ESI EVPN-IFL IP prefix routes adheres to the following conditions:

- Advertises the IP A-D per ES routes with the anycast MH flag set and anycast VTEP included
- Suppresses IP A-D per EVI routes
- Includes the EVPN router's MAC extended community in IP A-D per ES routes (required for VXLAN)
  - All egress leafs on the same anycast MH ES must advertise the same router MAC.
  - The router MAC is configured using the **system network-instance protocols vxlan router-mac-address** command, which overrides the system MAC as the inner source/destination MAC; only one is allowed per system.
  - This can be viewed in state under **inner-ethernet-header used-source-mac**.

Ingress leaf in anycast MH mode for non-zero ESI

The ingress leaf in anycast MH mode for non-zero ESI EVPN-IFL IP prefix routes adheres to the following conditions:

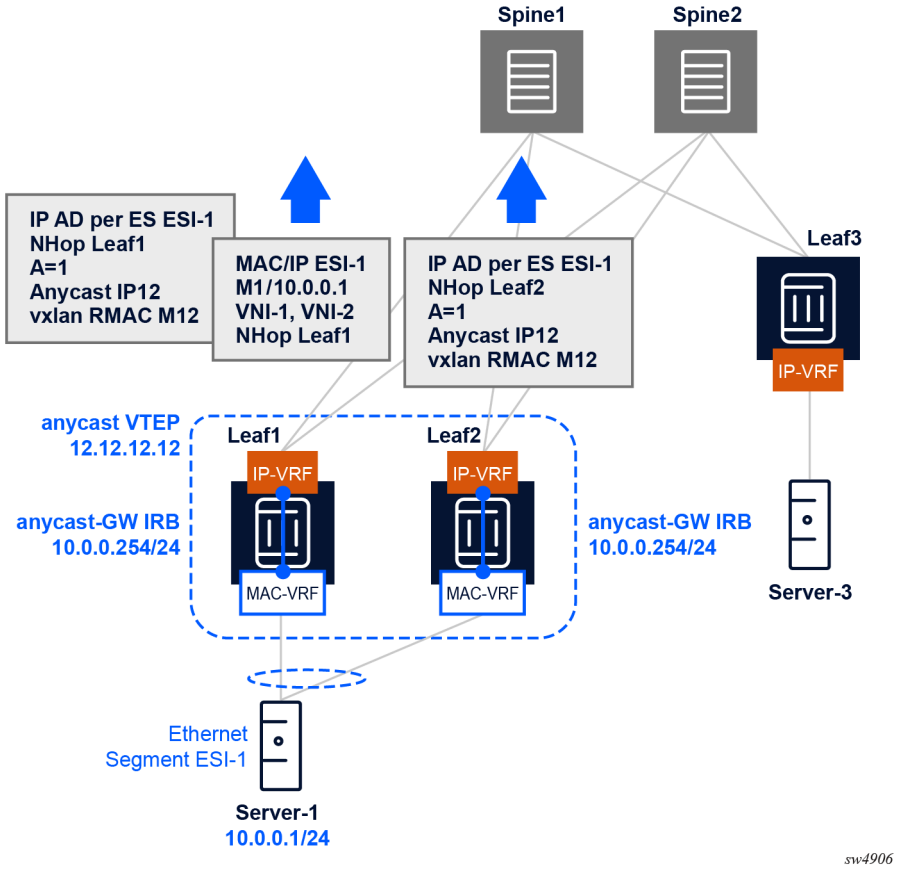
- Searches for an IP A-D per ES route with the same ESI in the same IP-VRF
- If found, it checks the anycast MH flag:
  - Flag not set – falls back to legacy IP aliasing behavior
  - Flag set – programs the EVPN-IFL entry toward an ES destination using the anycast VTEP
- In all other cases, it follows the same IP A-D per ES procedures as the Layer 2 ingress NVE, including error handling.
- For Layer 3 services, A-D per ES consistency requires that all contributing IP A-D per ES routes advertise the same anycast VTEP (BGP Tunnel Encapsulation Attribute) and the same EVPN router's MAC extended community (GW-MAC). If the anycast VTEP address or GW-MAC differs across contributing routes, the ingress leaf enters mismatch mode and falls back to installing the route using the BGP NLRI next hops of the contributing routes. A differing VNI across IP prefix routes does not affect consistency; when the anycast VTEP address and GW-MAC are consistent, the anycast VTEP is still used and the VNI is taken from the best IP prefix route.

4.11.2.2.1 Multi-homed non-zero ESI EVPN-IFL-HOST routes

Anycast MH can be applied to non-zero Ethernet Segment Identifier (ESI) EVPN-IFL-HOST routes, as shown in the following figure.

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

Figure 44: Anycast multi-homing for non-zero ESI EVPN-IFL-HOST routes



Configuring anycast multi-homing for non-zero ESI

The egress leafs attached to the anycast MH Ethernet Segments (Leaf1 and Leaf2 in the preceding figure) are configured with the **ethernet-segment anycast-multi-homing** command.

```
--{ * candidate shared default }--[  ]--
# info with-context system network-instance
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 2 {
                            ethernet-segment esi1 {
                                anycast-multi-homing {
                                    ip-address 12.12.12.12
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```


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

Egress leaf in anycast multi-homing mode for non-zero ESI

The egress leafs attached to the anycast MH Ethernet Segments (Leaf1 and Leaf2 in the preceding figure):

- Advertise both sets of Ethernet A-D per ES and IP A-D per ES routes with the anycast MH mode flag and the anycast VTEP
- Suppress the advertisement of both Ethernet A-D per EVI and IP A-D per EVI routes for ES configured in anycast MH mode
- Include the EVPN router's MAC extended community along with the IP A-D per ES routes (required for VXLAN)
  - When advertised with the IP A-D per ES routes, the EVPN router's MAC extended community must contain the same MAC address value in all the IP A-D per ES routes advertised by all the egress leafs attached to the same anycast MH ES.
  - The router MAC used for anycast MH in MAC/IP routes (EVPN-IFL-HOST routes) and IP A-D per ES routes is configured using the **system network-instance vxlan router-mac-address <mac>** command and overrides the system MAC used in routed VXLAN interfaces as inner **source-mac** and **router-mac** in the EVPN routes. The configured router's MAC is used as the inner source MAC address of the sent packets and is expected as the inner destination MAC address of the received packets on termination.
  - Only one **router-mac** per system can be configured.
  - The **router-mac** is exposed in the state for routed VXLAN interfaces under **inner-ethernet-header used-source-mac <mac>**.

Ingress leaf in anycast multi-homing mode for non-zero ESI

When receiving and importing an EVPN MAC/IP advertisement route with a non-zero ESI, the ingress leaf searches for an IP A-D per ES route with the same ESI imported into the same IP-VRF, and then checks whether the anycast MH flag is set.

- Flag not set – ingress NVE follows the IP aliasing procedures supported before this feature
- Flag set – ingress NVE programs the EVPN-IFL-HOST entry (from the MAC/IP advertisement route with two VNIs) to be associated with the ES destination that uses an anycast VTEP.

Other than the preceding conditions, the same procedures can be applied to IP A-D per ES routes as those defined for Ethernet A-D per-ES routes in the Layer 2 ingress NVE procedures, including all error-handling cases.

For Layer 3 services, IP A-D per-ES consistency requires that all contributing IP A-D per ES routes advertise the same anycast VTEP (BGP Tunnel Encapsulation Attribute) and the same EVPN router's MAC extended community (GW-MAC). If the anycast VTEP address or GW-MAC differs across contributing routes, the ingress leaf enters mismatch mode and falls back to installing the route using the BGP NLRI next hops of the contributing routes. A differing VNI across MAC/IP advertisement routes does not affect consistency; when the anycast VTEP address and GW-MAC are consistent, the anycast VTEP is still used and the VNI is taken from the best MAC/IP advertisement route.

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

5 IP-VPN services

Note: IP-VPN services are supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

This chapter describes the components of IP-VPN services on SR Linux.

5.1 Configuring IP-VPN services

IP-VPN services use a combination of MP-BGP and MPLS to distribute IPv4/v6 routing information and provide Layer 3 VPN services.

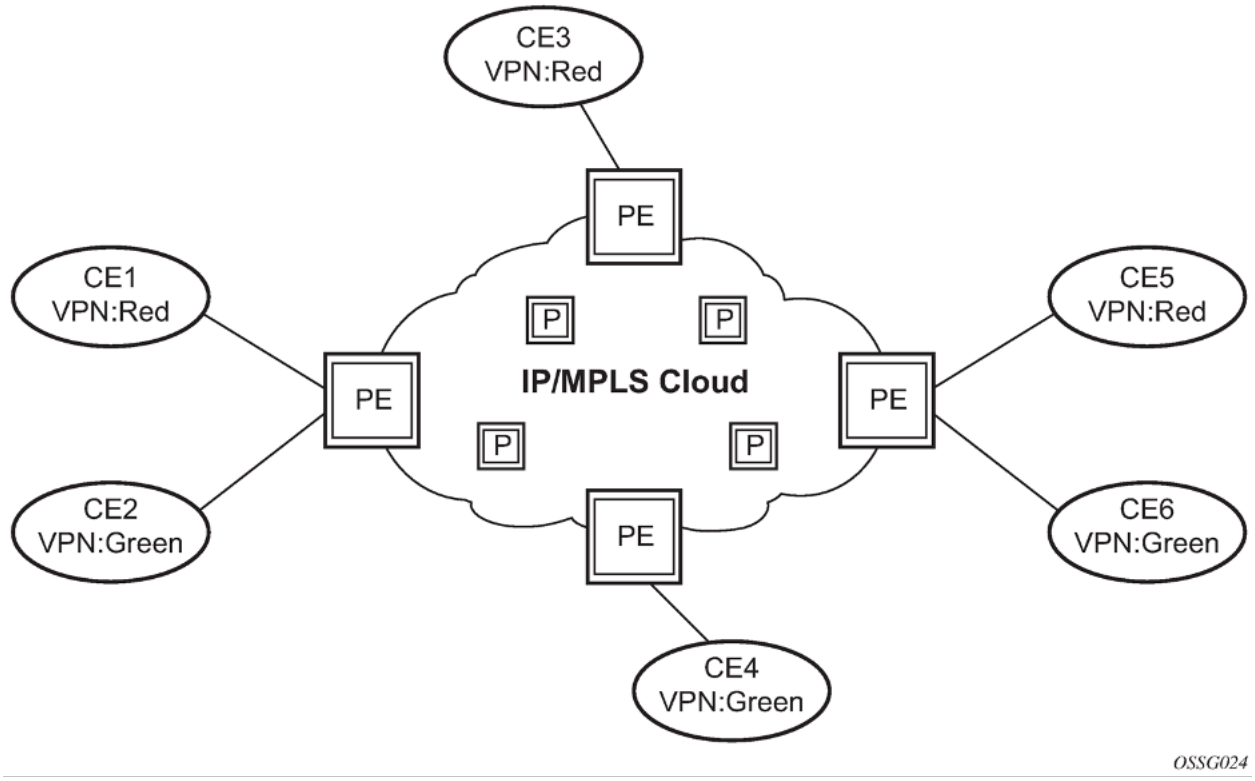
Each IP-VPN consists of a set of customer end points connected to one or more PE routers. Each associated PE router maintains a separate IP forwarding table for each IP-VPN instance. Additionally, the PE routers exchange the routing information configured or learned from all customer sites via MP-BGP peering. Each route exchanged via the MP-BGP protocol includes a Route Distinguisher (RD), which identifies the IP-VPN association and handles any potential IP address overlap.

Multi-Protocol BGP (MP-BGP) is used to exchange the routes of a particular VPN among the PE routers that are attached to that VPN. This route exchange is done in a way that ensures that routes from different VPNs remain distinct and separate, even if two VPNs have an overlapping address space. When BGP distributes a VPN route it also distributes an MPLS label for that route to identify the advertising IP-VPN instance.

Before a customer data packet travels across the service provider's backbone, it is encapsulated with the MPLS label that corresponds, in the customer's IP-VPN, to the route that best matches the packet's destination address. The MPLS packet is further encapsulated with one or more MPLS labels corresponding to the resolving MPLS path to deliver the packet to the intended egress PE router. The following figure displays an IP-VPN network diagram example.

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

Figure 45: IP Virtual Private Network



Example: IP-VPN configuration

The following is an example of an IP-VPN configuration. As a prerequisite, a label block must be configured using the **system mpls services network-instance dynamic-label-block** <*name*> configuration. See MPLS transport for EVPN services for more information about the label block required for network instances.

```
--{ * candidate shared default }--[  ]--
A:srl1# info with-context system mpls
    system {
        mpls {
            services {
                network-instance {
                    dynamic-label-block label-block-1
                }
             }
        }
    }
```

```
--{ * candidate shared default }--[  ]--
A:srl1# info with-context interface ethernet-1/2
    interface ethernet-1/2 {
        admin-state enable
        subinterface 1 {
            type routed
            admin-state enable
            ipv4 {
                address 10.30.30.1/24 {
```


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

```
                }
            }
        }
    }
A:srl1# info with-context network-instance Base
    network-instance Base {
        type default
        protocols {
            bgp {
                admin-state enable
                autonomous-system 65550
                router-id 10.10.10.1
                afi-safi l3vpn-ipv4-unicast {
                    admin-state enable
                }
                group base-group {
                }
                neighbor 10.10.10.2 {
                    peer-group base-group
                }
            }
        }
    }

--{ * candidate shared default }--[  ]--
A:srl1# info with-context network-instance ip-vrf-red
    network-instance ip-vrf-red {
        type ip-vrf
        interface ethernet-1/2 {
            interface-ref {
                interface ethernet-1/2
                subinterface 1
            }
        }
        protocols {
            bgp-ipvpn {
                bgp-instance 1 {
                    admin-state enable
                    ecmp 8
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [
                                ldp
                                sr-isis
                            ]
                        }
                    }
                }
            }
            bgp {
                admin-state enable
                autonomous-system 65551
                router-id 10.10.10.1
                afi-safi ipv4-unicast {
                    admin-state enable
                }
                group ip-vrf-red-peers {
                    admin-state enable
                    afi-safi ipv4-unicast {
                    }
                }
                neighbor 10.10.10.3 {
                    peer-group ip-vrf-red-peers
                    }
```


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

```
            }
        }
        bgp-vpn {
            bgp-instance 1 {
                route-distinguisher {}
                route-target {}
            }
        }
    }
```


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

6 BGP VPN Services

This chapter describes the components of various BGP VPN services on SR Linux.

6.1 BGP fast reroute in IP-VRF network instances

Note: This feature is supported on 7250 IXR Gen 3, 7250 IXR Gen 2, and 7250 IXR Gen 2c+ platforms.

BGP fast reroute (FRR), or Edge Prefix Independent Convergence (PIC), combines indirection techniques in the forwarding plane and pre-computation of BGP backup paths in the control plane to support fast reroute of BGP traffic around unreachable or failed BGP next hops.

When BGP FRR is enabled, BGP attempts to program a backup path for every IPv4 or IPv6 non-ECMP prefix in the route table that is derived from imported EVPN or IP-VPN routes.

For more information about BGP fast reroute, see the *SR Linux Routing Protocols Guide*.

Configuring BGP fast reroute

When **backup-paths install true** is configured, BGP finds and programs a backup path for every non-ECMP prefix within the scope of the command. A BGP-VPN route is a backup path candidate if it is valid, its BGP next-hop is reachable and its BGP next-hop is different than the BGP next-hop of the primary path. The backup path is the best of the candidate paths and selected using the same BGP decision process and tie-breaks that are applicable to primary path selection.

The following shows the configuration of BGP FRR for IPv4 VPN routes. For configuration that applies to IPv6 VPN routes, use the **ipv6-unicast** command.

```
--{ +* candidate shared default }--[  ]--
A:srl1# info network-instance test protocols
    network-instance ip-vrf {
        protocols {
            bgp-vpn {
                backup-paths {
                    ipv4-unicast {
                        install true
                    }
                }
            }
        }
    }
```

6.2 Inter-domain VPN services

This chapter describes the different inter-domain VPN services on SR Linux.

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

6.2.1 Multi-instance IP-VRF

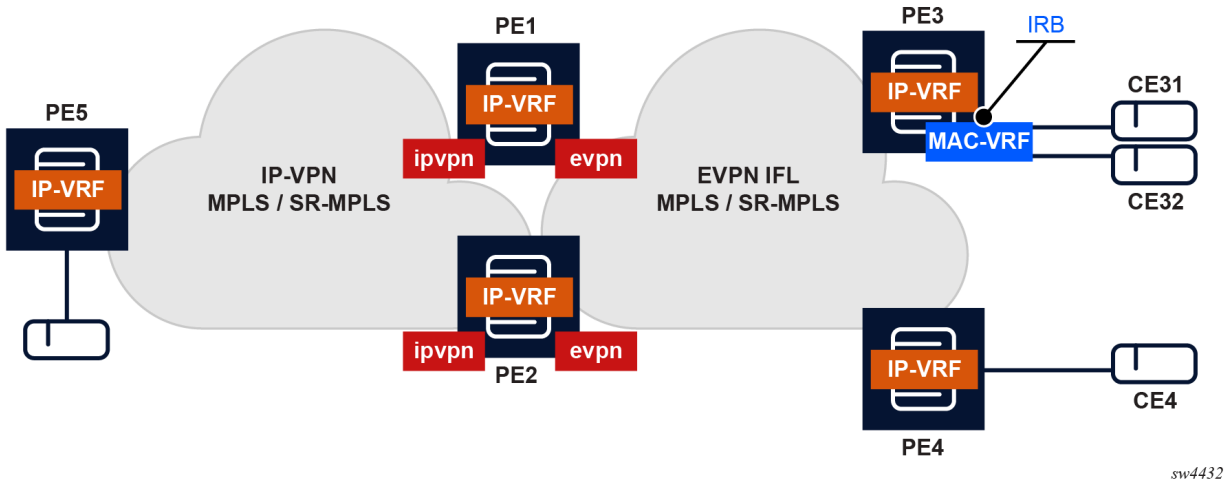
Note: Multi-instance IP-VRF is supported on 7250 IXR platforms.

Multi-instance IP-VRF refers to an IP-VRF network instance that supports multiple instances of BGP-VPN (a generalization of BGP families that are used for VPN services, EVPN, or IP-VPN). Multi-instance IP-VRF is configured using the command **network-instance protocols bgp-vpn bgp-instance**.

Multi-instance IP-VRFs are used on service gateway PE routers that stitch multiple domains together, in accordance with draft-ietf-bess-evpn-ipvpn-interworking. These domains can use different BGP Layer 3 families or different tunnels in the data path.

The following figure illustrates PE1 and PE2 as service gateways that connect an EVPN-IFL domain to an IP-VPN domain. PE1 and PE2 use a single IP-VRF with two BGP-VPN instances.

Figure 46: Multi-instance IP-VRF structure



The following example shows the configuration of PE1.

```
--[  ]--
A:pe1# info with-context network-instance IP-VRF1
  network-instance IP-VRF1 {
    type ip-vrf
    interface lo0.1 {
    }
    protocols {
      bgp-ipvpn {
        bgp-instance 1 {
          dpath-domain-id 64500:2
          mpls {
            next-hop-resolution {
              allowed-tunnel-types [ ldp sr-isis ]
            }
          }
        }
      }
      bgp-vpn {
        bgp-instance 1 {
```


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

```
          route-distinguisher { rd 1:1 }
          route-target {
            export-rt target:64500:10
            import-rt target:64500:10
          }
        }
        bgp-instance 2 {}
      }
    }
  }
```

The following shows the configuration of PE2.

```
--[  ]--
A:pe2# info with-context network-instance IP-VRF1
  network-instance IP-VRF1 {
    type ip-vrf
    interface lo0.1 {}
    protocols {
      bgp-evpn {
        bgp-instance 2 {
          dpath-domain-id 64500:2
          encapsulation-type mpls
          evi 10
          mpls {
            next-hop-resolution {
              allowed-tunnel-types [ bgp te-policy-sr-policy-mpls-uncolored ]
            }
          }
        }
      }
      bgp-ipvpn {
        bgp-instance 1 {
          dpath-domain-id 64500:2
          mpls {
            next-hop-resolution {
              allowed-tunnel-types [ ldp sr-isis ]
            }
          }
        }
      }
      bgp-vpn {
        bgp-instance 1 {
          route-distinguisher { rd 2:2 }
          route-target {
            export-rt target:64500:10
            import-rt target:64500:10
          }
        }
        bgp-instance 2 {}
      }
    }
  }
```

Up to two BGP instances are supported in the same network instance of type **ip-vrf**. One BGP instance is used for EVPN-IFL and the other BGP instance is used for IP-VPN.

If the same route is received on the same IP-VRF via different BGP owners, such as IP-VPN, EVPN-IFL, or BGP PE-CE, the selection and combined ECMP set is completed based on the best path selection for EVPN IFL routes. For more information, see Best path selection for EVPN IFL routes.

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

D-PATH is configured on each BGP instance to prevent control plane loops. For more information, see Domain path (D-PATH) attribute for inter-domain connectivity.

6.2.2 Multi-instance EVPN-IFL (MPLS-VXLAN)

Note: Multi-instance EVPN-IFL (MPLS-VXLAN) is supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

Multi-instance EVPN-IFL (MPLS-VXLAN) enables an IP-VRF network instance to host two BGP-EVPN instances configured using the **network-instance protocols bgp-evpn bgp-instance** command. One instance uses MPLS encapsulation, while the other uses VXLAN encapsulation. This capability aligns with the EVPN specification, *draft-ietf-bess-evpn-ipvpn-interworking*, which defines a Layer 3 interworking solution within a single integrated IP-VRF.

The feature is intended for service gateway PE routers that interconnect EVPN-VXLAN and EVPN-MPLS domains. Both BGP-EVPN instances share the same BGP owner. Route redistribution between the instances is enabled by configuring the **allow-export** parameter in the **network-instance protocols bgp-vpn** context.

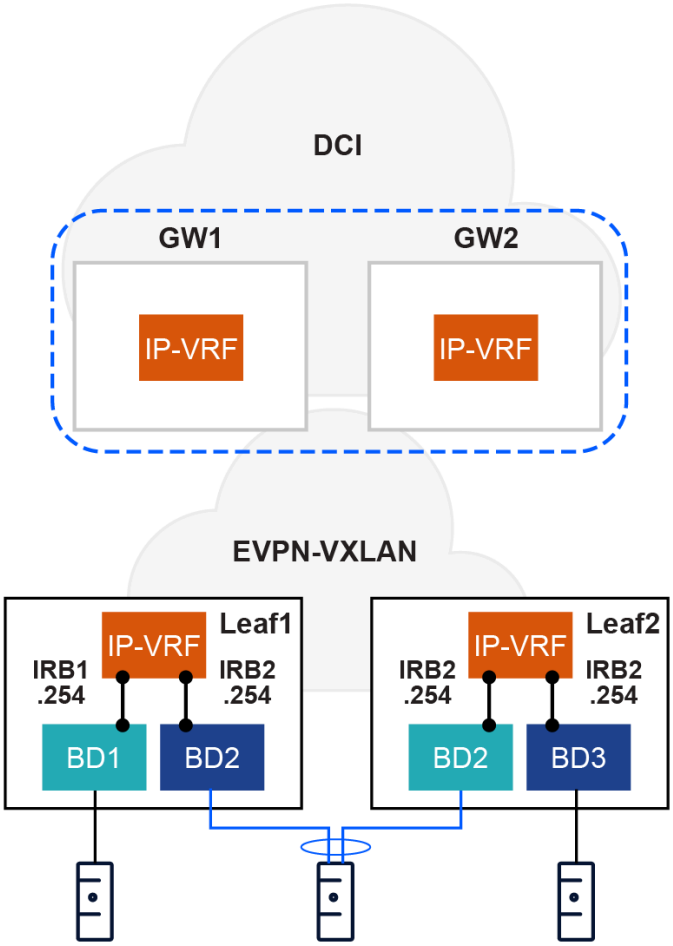
To enable multi-instance EVPN-IFL, configure two BGP-EVPN instances within the same IP-VRF. One operating with MPLS encapsulation and the other with VXLAN encapsulation. When **allow-export** is enabled, IFL and RT-2 host routes received in one BGP-EVPN instance are re-exported into the other instances.

Multi-homing mode is configured independently for each BGP-EVPN instance using the **bgp-evpn bgp-instance multi-homing-mode** command. One instance must operate in **access** mode and the other in **network** mode (default). Changing the multi-homing role requires reprogramming the instance. In multi-instance IP-VRF deployments, the **network** mode instance is responsible for Designated Forwarder (DF) election, Ethernet Segment (ES) procedures, processing IP AD per ES and AD per EVI routes, and supporting IP aliasing.

The following figure shows GW1 and GW2 acting as service gateways between EVPN-VXLAN and EVPN-MPLS domains. Each gateway uses a single IP-VRF containing two BGP-EVPN instances.

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

Figure 47: Multi-instance EVPN-IFL (MPLS-VXLAN) structure



The following implementation considerations apply:

- An IP-prefix or RT-2 host route can be received with route targets that match both instances.
- The same route target can be configured on both instances.
- D-PATH must be configured on each BGP-EVPN instance to prevent control-plane loops. For more information, see Domain path (D-PATH) attribute for inter-domain connectivity.

The following example shows the configuration of GW1.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance IP-VRF
    network-instance IP-VRF {
        type ip-vrf
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
```


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

```
                    evi 1
                    ecmp 4
                    dpath-domain-id 1:1
                    multi-homing-mode access
                }
                bgp-instance 2 {
                    encapsulation-type mpls
                    evi 2
                    ecmp 4
                    dpath-domain-id 2:2
                    multi-homing-mode network
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [
                                ldp
                            ]
                        }
                    }
                }
            }
            bgp-vpn {
                allow-export {
                }
            }
        }
    }
```

6.2.3 Multi-instance MAC-VRF

Note: Multi-instance MAC-VRF is supported on 7250 IXR Gen 2c+ and 7250 IXR Gen 3 platforms.

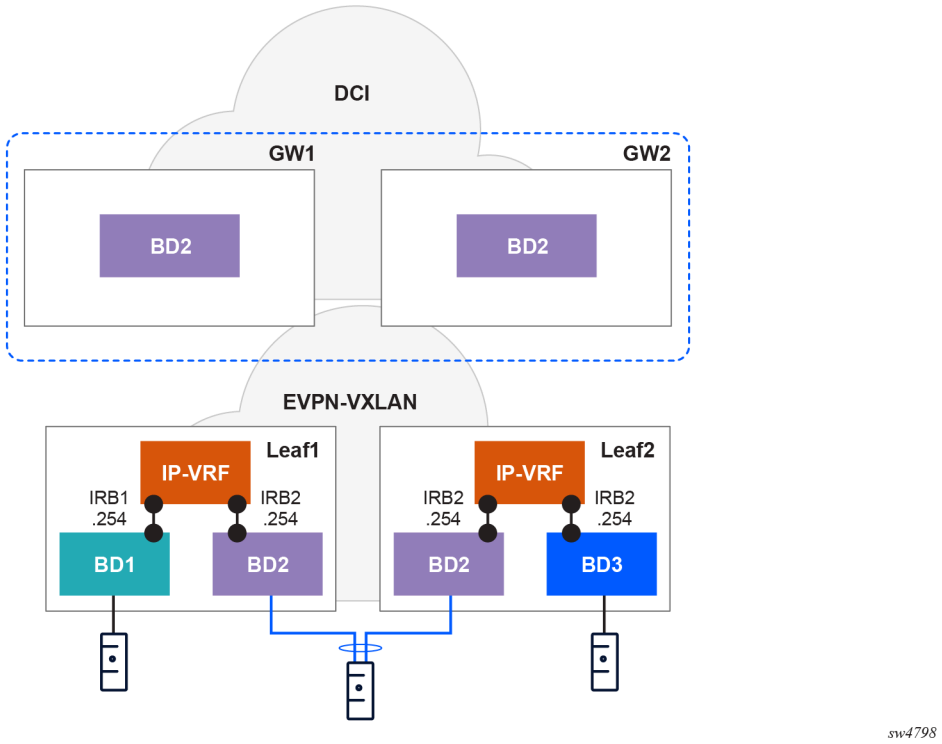
Multi-instance MAC-VRF refers to a MAC-VRF network instance that supports multiple instances of BGP-EVPN. Multi-instance MAC-VRF is configured using the **network-instance protocols bgp-evpn bgp-instance** command. Multi-instance MAC-VRFs are deployed on service gateways that interconnect broadcast domains across two different EVPN domains, as specified in RFC 9014. Each domain can operate its own IGP instance, BGP Autonomous System, or transport tunnel type.

Only a single BGP-EVPN encapsulation is supported per network instance. Only the combination of BGP-EVPN VXLAN and BGP-EVPN MPLS in the same MAC-VRF is allowed. When two BGP-EVPN instances are enabled in a MAC-VRF, no subinterfaces can be configured on the same MAC-VRF.

The following figure illustrates the multi-instance MAC-VRF structure in two service gateways - GW1 and GW2 - that interconnect an EVPN-VXLAN domain in the data center to an EVPN-MPLS domain in the WAN.

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

Figure 48: Multi-instance MAC-VRF structure



On MAC-VRFs with two BGP-EVPN instances, the received EVPN routes in one BGP-EVPN instance are processed as follows:

- EVPN A-D per ES and per EVI routes (type 1 routes) – consumed but not propagated to the other instance
- EVPN MAC/IP Advertisement routes (type 2 routes) – consumed and - if programmed in the MAC table - propagated to the other instance
- EVPN Inclusive Multicast Ethernet Tag routes (type 3 routes) – consumed but not propagated to the other instance
- EVPN Ethernet Segment routes (type 4 routes) – not imported because the MAC-VRF does not support local Ethernet Segments

EVPN MAC/IP routes are redistributed as follows:

- Only the best MAC/IP route is redistributed for a specified MAC and IP.
- In the case that multiple MAC/IP routes are received for the same MAC but with different IP addresses, all routes are redistributed to the other instance.
- EVPN MAC mobility and ARP/ND extended communities received on MAC/IP advertisement routes are propagated unchanged to the other instance.

The following considerations apply to multi-instance MAC-VRFs:

- As per RFC 9014, the configuration of the **vlan-aware-bundle-eth-tag ethernet-tag-id** command needs to be the same across the two instances.
- Multi-instance MAC-VRFs do not support proxy-ARP/ND or multicast snooping protocols.

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

- In a MAC-VRF with two BGP-EVPN instances, only one instance can be set to **multi-homing-mode network (default value)**. However, both instances can be set to **multi-homing-mode access**.
- Multi-instance MAC-VRF association is not allowed under eth-CFM.
- Connection point is not supported under multi-instance MAC-VRF.
- Static MAC configuration is not supported under multi-instance MAC-VRF.

Example: Configuration of MAC-VRF network instance BD2

The following example shows the configuration of the MAC-VRF network instance BD2.

```
--{ * candidate shared default }--[  ]--
A:admin@gw1# info with-context
    network-instance BD2 {
        type mac-vrf
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan0.1
                    evi 1
                    ecmp 2
                    multi-homing-mode access
                    internal-tags {
                        set-tag-set [
                            dc
                        ]
                    }
                    routes {
                        bridge-table {
                            inclusive-mcast {
                                originating-ip 11.12.11.12
                            }
                        }
                    }
                }
                bgp-instance 2 {
                    encapsulation-type mpls
                    evi 2
                    ecmp 2
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [
                                ldp
                                sr-isis
                            ]
                        }
                    }
                    routes {
                        bridge-table {
                            inclusive-mcast {
                                originating-ip 11.12.11.12
                            }
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                    route-distinguisher {
```


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

```
                        rd 100.11.12.1:1
                    }
                    route-target {
                        export-rt target:64500:1
                        import-rt target:64500:1
                    }
                }
                bgp-instance 2 {
                    route-distinguisher {
                        rd 100.11.12.2:1
                    }
                    route-target {
                        export-rt target:64500:1
                        import-rt target:64500:1
                    }
                }
            }
        }
    }
```

Multi-instance gateway redundancy

Multi-instance MAC-VRF service gateways are usually deployed in a redundant configuration, with two or more gateways operating as part of the same redundancy group. These gateways provide stitching between the VXLAN and MPLS domains for a specified broadcast domain. To avoid loops, the redundant gateways for a specified multi-instance MAC-VRF must be configured as follows:

- The EVPN-VXLAN instance must use the same **route-distinguisher** and **originating-ip** on all redundant gateways.
- The EVPN-MPLS instance must use the same **route-distinguisher** and **originating-ip** on all the gateways.

Remote leaf nodes receive equivalent IMET and MAC/IP routes from all redundant gateways, and BGP selects routes from only one of the gateways.

In addition, redundant gateways receive each other's IMET and MAC/IP advertisement routes. Gateway IMET routes are received on a gateway with an **originating-ip** and **route-distinguisher** matching the local **originating-ip** and **route-distinguisher** values, so EVPN multicast destinations are not created among the redundant gateways. BGP policies are needed to avoid control plane loops for MAC/IP Advertisement routes. These policies prevent gateways from redistributing MAC/IP Advertisement routes that are received from another gateway.

The following example shows the configuration of the MAC-VRF network instance BD1 on two redundant gateways, dc1gw1 and dc1gw2:

```
// dc1gw1 configuration
--{ candidate shared default }--[ network-instance BD1 ]--
A:admin@dc1gw1# info with-context
    network-instance BD1 {
        type mac-vrf
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan0.1
                    evi 1
                    ecmp 2
                    multi-homing-mode access
```


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

```
                    internal-tags {
                        set-tag-set [
                            dc
                        ]
                    }
                    routes {
                        bridge-table {
                            inclusive-mcast {
                                originating-ip 11.12.11.12
                            }
                        }
                    }
                }
                bgp-instance 2 {
                    encapsulation-type mpls
                    evi 2
                    ecmp 2
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [
                                ldp
                                sr-isis
                            ]
                        }
                    }
                    routes {
                        bridge-table {
                            inclusive-mcast {
                                originating-ip 11.12.11.12
                            }
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                    route-distinguisher {
                        rd 100.11.12.1:1
                    }
                    route-target {
                        export-rt target:64500:1
                        import-rt target:64500:1
                    }
                }
                bgp-instance 2 {
                    route-distinguisher {
                        rd 100.11.12.2:1
                    }
                    route-target {
                        export-rt target:64500:1
                        import-rt target:64500:1
                    }
                }
            }
        }
    }

// dc1gw2 configuration
--{ candidate shared default }--[ network-instance BD1 ]--
A:admin@dc1gw2# info with-context
    network-instance BD1 {
        type mac-vrf
        admin-state enable
        vxlan-interface vxlan0.1 {
```


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

```
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan0.1
                    evi 1
                    ecmp 2
                    multi-homing-mode access
                    internal-tags {
                        set-tag-set [
                            dc
                        ]
                    }
                    routes {
                        bridge-table {
                            inclusive-mcast {
                                originating-ip 11.12.11.12
                            }
                        }
                    }
                }
                bgp-instance 2 {
                    encapsulation-type mpls
                    evi 2
                    ecmp 2
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [
                                ldp
                                sr-isis
                            ]
                        }
                    }
                    routes {
                        bridge-table {
                            inclusive-mcast {
                                originating-ip 11.12.11.12
                            }
                        }
                    }
                }
            }
            bgp-vpn {
                bgp-instance 1 {
                    route-distinguisher {
                        rd 100.11.12.1:1
                    }
                    route-target {
                        export-rt target:64500:1
                        import-rt target:64500:1
                    }
                }
                bgp-instance 2 {
                    route-distinguisher {
                        rd 100.11.12.2:1
                    }
                    route-target {
                        export-rt target:64500:1
                        import-rt target:64500:1
                    }
                }
            }
        }
```


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

```
    }

// Routing policies to prevent loops (same configuration on both gateways)
--{ candidate shared default }--[ network-instance default protocols bgp neighbor * ]--
A:admin@dc1gw2# info
    neighbor 100.1.0.1 { // peer in EVPN-VXLAN domain
        peer-as 64500
        peer-group overlay
        export-policy [
            allow-only-vxlan-and-add-SOO-DC1
        ]
        import-policy [
            drop-SOO-DC1
        ]
    }
    neighbor 100.1.0.11 {  // peer in EVPN-MPLS domain
        peer-as 64500
        peer-group overlay
        export-policy [
            allow-only-mpls-and-add-SOO-DC1
        ]
        import-policy [
            drop-SOO-DC1
        ]
    }

--{ candidate shared default }--[ routing-policy ]--
A:admin@dc1gw2# info with-context
    routing-policy {
        extended-community-set SOO-DC1 {
            member [
                origin:1:1
            ]
        }
        extended-community-set bgp-tunnel-encap:MPLS {
            member [
                bgp-tunnel-encap:MPLS
            ]
        }
        extended-community-set bgp-tunnel-encap:VXLAN {
            member [
                bgp-tunnel-encap:VXLAN
            ]
        }
        tag-set dc {
            tag-value [
                1
            ]
        }
        tag-set wan {
            tag-value [
                2
            ]
        }
        policy allow-only-mpls-and-add-SOO-DC1 {
            statement 10 {
                match {
                    family [
                        evpn
                    ]
                    internal-tags {
                        tag-set [
                            dc
                        ]
```


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

```
                }
                bgp {
                    extended-community {
                        extended-community-set bgp-tunnel-encap:VXLAN
                    }
                }
            }
            action {
                policy-result reject
            }
        }
        statement 20 {
            match {
                family [
                    evpn
                ]
            }
            action {
                policy-result accept
                bgp {
                    extended-community {
                        operation add
                        referenced-sets [
                            SOO-DC1
                        ]
                    }
                }
            }
        }
    }
    policy allow-only-vxlan-and-add-SOO-DC1 {
        statement 10 {
            match {
                family [
                    evpn
                ]
                bgp {
                    extended-community {
                        extended-community-set bgp-tunnel-encap:MPLS
                    }
                }
            }
            action {
                policy-result reject
            }
        }
        statement 20 {
            match {
                family [
                    evpn
                ]
            }
            action {
                policy-result accept
                bgp {
                    extended-community {
                        operation add
                        referenced-sets [
                            SOO-DC1
                        ]
                    }
                }
            }
        }
```


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

```
        }
        policy drop-SOO-DC1 {
            statement 10 {
                match {
                    family [
                        evpn
                    ]
                    bgp {
                        extended-community {
                            extended-community-set SOO-DC1
                        }
                    }
                }
                action {
                    policy-result reject
                }
            }
        }
    }
```

Source VTEP security in anycast redundant gateway scenarios

The VXLAN implementation on Broadcom chipsets requires the source VTEP IP address to be matched in ingress tables before a VXLAN packet is terminated. As a result, a node must know the source VTEP IP address to successfully terminate incoming VXLAN traffic.

In a single-instance EVPN VXLAN network, this requirement is normally satisfied because all VXLAN VTEPs are learned through IMET routes, which are exchanged among all nodes before traffic is forwarded. However, in a multi-instance EVPN VXLAN deployment with multiple redundant gateways, a leaf node may never learn the VTEP IP address of one of the anycast gateways, even though that gateway may still send traffic to it. This is a result of configuring the same **route-distinguisher** and **originating-ip** in all the redundant gateways for a specific MAC-VRF.

In gateway redundancy designs that include Layer 3 services, this issue does not typically arise, because Layer 3 services do not rely on configuring the same **route-distinguisher** and **originating-ip** in the same way, and leaf nodes learn all source VTEPs.

However, if the gateways provide only Layer 2 services, leaf nodes may learn the VTEP of only one of the gateways in the redundancy group. In this case, VXLAN packets sourced from the other gateway are not terminated.

If only Layer 2 services are configured on the redundant gateways, it is recommended to configure a dummy MAC-VRF service on both the gateways and the local leaf nodes. This dummy service:

- uses different RDs on each gateway
- does not carry real traffic
- exists to ensure that IMET routes are exchanged, allowing leaf nodes to learn all source VTEPs of the anycast gateways

When the above configuration is implemented in the redundant service gateways, all leaf nodes learn the service gateway VTEPs, and traffic is forwarded normally.

6.2.4 Inter-domain EVPN IFL and IP-VPN in a single IP-VRF BGP instance

Note: This feature is supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, 7250 IXR Gen 3, and 7220 IXR-D*x* platforms.

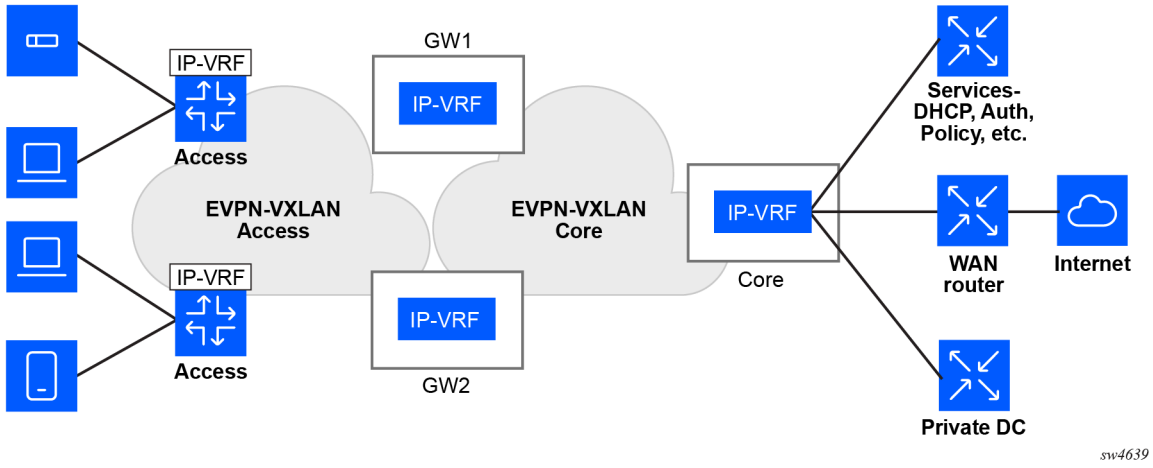
© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

This inter-domain EVPN IFL/IP-VPN solution is applicable to IP-VRF network instances that use a single BGP instance (enabled through either BGP-EVPN or BGP IP-VPN). This solution can be used to interconnect two domains seamlessly through a gateway.

Gateways provide IGP isolation between domains so that nodes in each domain do not need end-to-end visibility of all loopbacks in the network; they only need tunnels to the gateways. In the overlay layer, gateways provide prefix aggregation or advertise a single default route to the access domain to contain the number of routes that the access nodes need to program in their forwarding information bases (FIBs).

The following figure illustrates an example of how this inter-domain solution for Layer 3 services can be applied. In this scenario, there are two domains: an EVPN-VXLAN access domain and an EVPN-VXLAN core domain. Each domain runs its own IGP instance and loopback IP addresses are not leaked between them. This minimizes the number of VXLAN tunnels required on the access routers. Gateways GW1 and GW2 are configured with an IP-VRF that has EVPN-VXLAN IFL enabled, allowing them to redistribute EVPN IP Prefix routes between the two domains.

Figure 49: Inter-domain Layer 3 solution



The following example shows the configuration of an IP-VRF in the gateways.

```
--{ * candidate shared default }--[ network-instance IP-VRF ]--
A:root@srl1# info with-context
    network-instance IP-VRF {
        type ip-vrf
        vxlan-interface vxlan0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    vxlan-interface vxlan0.1
                }
            }
            bgp-vpn {
                allow-export {
                }
                bgp-instance 1 {
                }
            }
        }
    }
```


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

The configuration of **allow-export** is only supported in IP-VRF network instances and provides the following functionalities:

- Received EVPN IFL or IP-VPN routes are re-exported to the configured BGP peers. In IP-VRF network instances that contain both BGP-EVPN and BGP-IPVPN instances, EVPN IFL or IP-VPN routes are automatically re-exported to the BGP-IPVPN instance without the **allow-export** command.
- Received EVPN IFL routes are imported and re-advertised using the local RD/RT, next-hop, or mac-nh.
- Local label or VNI is advertised with the routes, which preserves the VNI in VXLAN routes end-to-end.
- Non-EVPN BGP attributes are propagated.
- In EVPN IFL, the preceding functionality is supported regardless of the encapsulation in the network instance (VXLAN or MPLS).

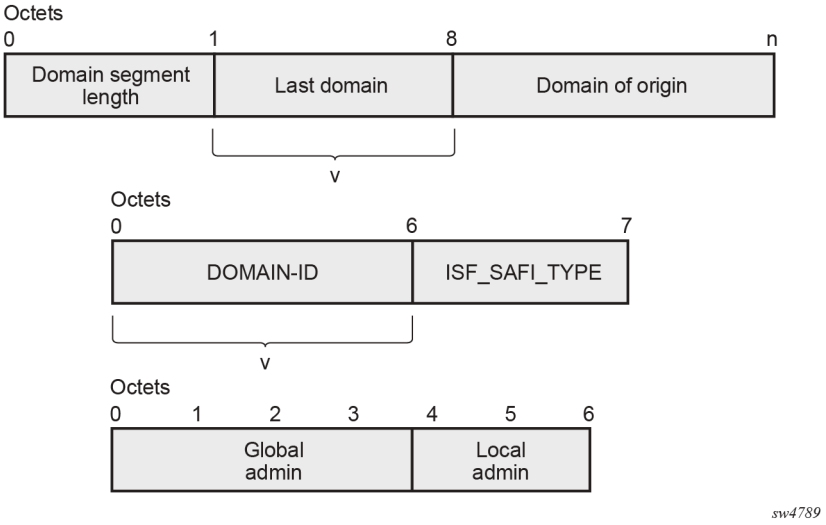
Note: The **allow-export** command should be used with caution, as it causes the router to export IP-VRF imported routes to all BGP peers, including the peers from which the routes were originally received. To prevent re-advertising routes back to their source peers and avoid routing loops in scenarios with redundant gateways, routing policies must be applied to filter unwanted advertisements.

6.2.5 Domain path (D-PATH) attribute for inter-domain connectivity

The BGP domain path (D-PATH) attribute is required in networks that expand multiple BGP, IP-VPN, or EVPN domains, as defined in *draft-ietf-bess-evpn-ipvpn-interworking*.

Similar to AS_PATH, D-PATH is composed of a sequence of domain segments. The following figure graphically represents each domain segment.

Figure 50: D-PATH attribute



Where:

- Each domain segment comprises of the *domain_segment_length*, *domain_segment_value*, where the domain segment value is a sequence of one or more domains.

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

- Each domain is represented by *DOMAIN-ID:ISF_SAFI_TYPE*, where the newly added domain, which is added by a gateway, is always prepended to the left of the last existing last.
- The supported ISF_SAFI_TYPE values are:
  - 0 = Local ISF route
  - 70 = evpn
  - 128 = safi 128 (IP-VPN domains)
- Labeled unicast IP or BGP PE-CE routes do not support D-PATH.
- The D-PATH attribute is only modified by a gateway and not by an ABR/ASBR or RR. A gateway is defined as a PE where an IP-VRF is instantiated, and that IP-VRF advertises or receives routes from multiple BGP owners (for example, EVPN IFL and BGP IP-VPN), multiple instances of the same owner (for example, IP-VRF with two BGP IP-VPN instances), or local routes.

Example:

Suppose a router receives prefix P in an EVPN-IFL instance with the following D-PATH from neighbor N.

```
+----------+------------+
|Seg Len=1 | 65000:1:128|
+----------+------------+
```

If the router imports the route in IP-VRF-1, BGP-EVPN instance with domain 65000:2, it re-advertises it to its BGP IP-VPN MPLS instance as follows.

```
+----------+----------+-----------+
|Seg Len=2 |65000:2:70|65000:1:128|
+----------+----------+-----------+
```

If the router imports the route in IP-VRF-1, BGP-EVPN instance with domain 65000:3, it re-advertises it to its BGP-EVPN MPLS instance as follows.

```
+----------+----------+-----------+
|Seg Len=2 |65000:3:70|65000:1:128|
+----------+----------+-----------+
```

If the router does not import the route in any service, but it re-advertises it to another neighbor M, as a result of the configuration of cluster (RR), ASBR, or NHS-RR, it re-advertises it with no changes.

```
+----------+------------+
|Seg Len=1 | 65000:1:128|
+----------+------------+
```

Note: In SR Linux, instead of advertising multiple domains in the same domain segment, multiple segments with one domain each are advertised. On reception, an SR Linux router can accept both ways: multiple segments or a single segment with multiple domains.

6.2.5.1 D-PATH configuration

In order for a router to add a D-PATH attribute or a new segment to the existing D-PATH, a **domain-id** needs to be configured. The D-PATH is modified on transmission or processed on reception based on the local network instance configuration. The **domain-id** is configured per BGP instance and the

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

ISF_SAFI_TYPE automatically derived from the instance type that imported the original route. The **domain-id** is configured at the service BGP instance level as a 6 byte value that includes a global admin value and a local admin value, for example 65000:1.

The following examples show how the **domain-id** can be configured:

- IP-VRF BGP-EVPN MPLS and VXLAN instances (EVPN-IFL):

  network-instance protocols bgp-evpn bgp-instance dpath-domain-id *global-field:local-field*

- IP-VRF BGP IP-VPN MPLS:

  network-instance protocols bgp-ipvpn bgp-instance dpath-domain-id *global-field:local-field*

- IP-VRF network instance level for local routes (direct, static, IGP routes):

  network-instance protocols bgp-vpn local-routes-dpath-domain-id *global-field:local-field*

  EVPN IFF routes are considered local routes for the purpose of adding D-PATH when readvertising as an IP-VPN/EVPN-IFL route. The **domain-id** for local routes is configured under BGP-VPN because it only applies to VPN routes.

The following consideration apply to the configuration of **domain-id**:

- The value of a **domain-id** cannot be 0:0.
- If **local-routes-dpath-domain-id** is not configured, the local routes are advertised into the BGP instances with no D-PATH.
- If a network instance BGP instance is not configured with a **domain-id**:
  - Routes imported in that instance are re-advertised in a different instance without modifying the D-PATH.
  - Routes exported in that instance are advertised with the D-PATH modified to include the **domain-id** of the instance that imported the route in the first place.
- There is no check for the uniqueness of the **domain-id** within the network instance or across network instances.
- Modifying the **domain-id** list initiates a route-refresh for all the address families associated with the network instance.

6.2.5.2 D-PATH and loop detection

When an EVPN IFL or IP-VPN route with a D-PATH containing a locally-configured **domain-id** on the IP-VRF is received where the route is imported, the route becomes inactive. Additionally, the `rib-in-out rib in-post/local` state of the route shows `invalid-reason domain-path-loop true`. The local RIB, `rib-in-pre`, and `rib-in-post` shows a list of network instances that detected a loop caused by the presence of a local **domain-id** in the imported route.

It is possible that a route has a looped domain for one IP-VRF and not for another. BGP installs the route in the IP-VRF that does not have loop and does not install the route in the IP-VRF that has the loop.

The `invalid-reason domain-path-loop true` output is exposed for routes that are not imported in any IP-VRF caused by matching the local **domain-id**s in the importing IP-VRFs.

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

6.2.5.3 D-PATH and best path selection

D-PATH is considered for BGP best path selection, as per draft-ietf-bess-evpn-ipvpn-interworking. In SR Linux, the best path selection process includes D-PATH in the tie-breaker sequence, as shown in the following list:

- Valid route preferred over invalid route
- Lowest route preference
- Highest Local Pref
- **Shortest D-PATH**
- Shortest AS Path
- Lowest Origin
- Lowest MED
- EBGP preferred over IBGP
- Lowest next-hop cost (break off for ECMP paths)
- Lowest next-hop type (TTM resolution wins over route table resolution)
- Route owner comparison (BGP preferred over BGP-label preferred over BGP-VPN)
- Lowest peer router ID
- Lowest peer IP address
- Route owner comparison (BGP EVPN IFL host preferred over BGP EVPN preferred over BGP IP-VPN)

As D-PATH is introduced in networks, not all the PEs will support D-PATH for BGP path selection. To guarantee compatibility in brown field scenarios, the command **network instance(default) protocols bgp best-path-selection ignore-domain-path-length <boolean>** is introduced, where:

- The command is set to false by default and the shorter D-PATH length is considered a tie breaker rule in the best path selection criteria for received routes with the same or different route distinguisher. The D-PATH length tiebreaker rule is compared immediately after the Local Preference comparison.
- When the command is set to true, the D-PATH is ignored for the purpose of best path selection.
- A tie-break reason in the BGP RIB model (domain-path-length) is displayed when the route is not active because of the D-PATH length.

6.2.6 Next-hop-self route reflector and inter-AS option B



Note: Next-hop-self route reflectors and inter-AS option B are supported on 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

Next-hop-self route reflectors (NHS-RR) are used in BGP networks to reduce the need for fully meshed iBGP connections within a single autonomous system (AS). In a fully meshed AS, iBGP routers do not advertise routes to their neighbors, while NHS-RR allows a route reflector (RR) to advertise learned iBGP routes to iBGP neighbors with its own address as the next hop. This feature connects different domains within the same AS and improves the overall scale.

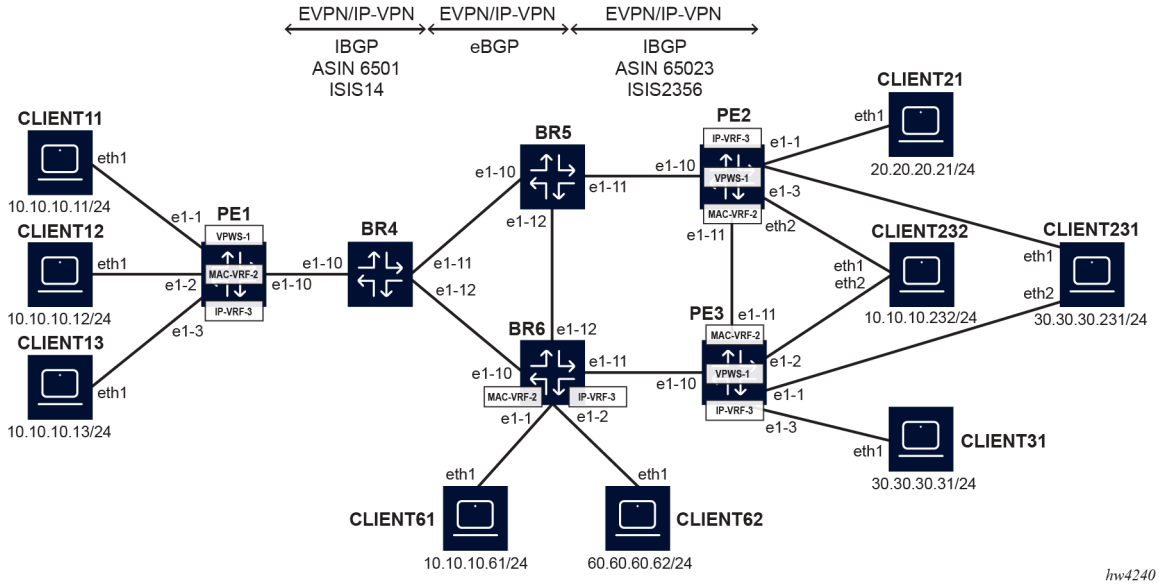
© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

Inter-AS option B is a method for interconnecting VPN sites located in different ASes. Using this method, autonomous system border routers (ASBRs) are directly connected and routes are exchanged on a single interface.

EVPN IFL and VPN-IPv4/IPv6 routes both support NHS-RR capability and inter-AS option B.

The following figure shows the configuration of NHS-RR and inter-AS option B:

Figure 51: EVPN and IP-VPN services on inter-AS option B



The preceding figure shows two different ASes that are connected via inter-AS option B ASBRs br4, br5, and br6. This method allows for the extension of EVPN and IP-VPN services across different MPLS or segement routing MPLS domains without providing services on border routers.

Egress PEs advertise EVPN and IP-VPN routes to adjacent border routers.

Border routers carry out the following functions:

1. Import and redistribute routes to the remote border routers or local PEs where the addresses of the remote border routers are used as the next hops and their own service MPLS labels are used.
2. Program a label swap operation so that the ingress traffic service label is looked up and packets are forwarded with a new service label.

The following example shows the configuration of the default network instance of pe1 in Figure 51: EVPN and IP-VPN services on inter-AS option B, which is similar to the configuration of the other PEs in the figure:

```
--{ + candidate shared default }--[ network-instance default ]--
A:pe1# info
    type default
    interface ethernet-1/10.0 {
    }
    interface system0.0 {
    }
    protocols {
        bgp {
            admin-state enable
```


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

```
            autonomous-system 65001
            router-id 10.0.0.1
            bgp-label {
                bgp-ipvpn {
                    next-hop-resolution {
                        ipv4-next-hops {
                            tunnel-resolution {
                                allowed-tunnel-types [
                                    ldp
                                    sr-isis
                                ]
                            }
                        }
                    }
                }
            }
            isis {
                dynamic-label-block range-3-srgb
                instance i14 {
                    admin-state enable
                    instance-id 1
                    level-capability L2
                    iid-tlv true
                    net [
                        49.0001.0000.0000.0001.00
                    ]
                    trace-options {
                        trace [
                            adjacencies
                            interfaces
                            packets-all
                        ]
                    }
                    segment-routing {
                        mpls {
                            dynamic-adjacency-sids {
                                all-interfaces true
                            }
                        }
                    }
                    interface ethernet-1/10.0 {
                        circuit-type point-to-point
                        ipv4-unicast {
                            admin-state enable
                        }
                    }
                    interface system0.0 {
                        passive true
                        ipv4-unicast {
                            admin-state enable
                        }
                    }
                }
            }
        }
        segment-routing {
            mpls {
                global-block {
                    label-range range-2-srgb
                }
                local-prefix-sid 1 {
                    interface system0.0
                    ipv4-label-index 1
                }
```


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

```
            }
        }
    }
```

The following example shows the configuration of the default network instance of RR pe2 in Figure 51: EVPN and IP-VPN services on inter-AS option B:

```
--{ + candidate shared default }--[ network-instance default protocols bgp ]--
A:pe2# info
    admin-state enable
    autonomous-system 65023
    router-id 10.0.0.2
    bgp-label {
        bgp-ipvpn {
            next-hop-resolution {
                ipv4-next-hops {
                    tunnel-resolution {
                        allowed-tunnel-types [
                            ldp
                            sr-isis
                        ]
                    }
                }
                ebgp-default-policy {
                    import-reject-all false
                    export-reject-all false
                }
                afi-safi evpn {
                    admin-state enable
                    evpn {
                        keep-all-routes true
                        rapid-update true
                    }
                }
                afi-safi l3vpn-ipv4-unicast {
                    admin-state enable
                    l3vpn-ipv4-unicast {
                        keep-all-routes true
                        rapid-update true
                    }
                }
                group overlay-ibgp {
                    peer-as 65023
                    afi-safi evpn {
                    }
                    afi-safi l3vpn-ipv4-unicast {
                    }
                    route-reflector {
                        client true
                        cluster-id 2.2.2.2
                    }
                    timers {
                        connect-retry 1
                        minimum-advertisement-interval 1
                    }
                    trace-options {
                        flag update {
                            modifier detail
                        }
                    }
                }
                neighbor 10.0.0.3 {
                    peer-group overlay-ibgp
```


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

```
                }
                neighbor 10.0.0.5 {
                    peer-group overlay-ibgp
                }
                neighbor 10.0.0.6 {
                    peer-group overlay-ibgp
                }
            }
        }
    }
```

The following commands enable NHS-RR and inter-AS option B functionality on SR Linux:

- **inter-as-vpn**
- **next-hop-self-route-reflector**

These commands affect all EVPN routes and trigger `bgp_mgr` to swap the service label for all EVPN MPLS routes. These commands also change the next hop to self in all routes with MPLS encapsulation.

For border router configuration, IGP is not enabled in interfaces to other border routers.

The **inter-as-vpn true** command allows received EVPN/IP-VPN routes to be retained in the BGP RIB and propagated to any eBGP or iBGP peer. To ensure label allocation, a dynamic label block must be configured for border routers. Label allocation re-advertises a received route into the adjacent AS with a local network instance MPLS label and ensures an MPLS label swap operation is completed.

Configure a dynamic label block using the following configuration:

```
--{ * candidate shared default }--[  ]--
A:srl1# info with-context network-instance default
    network-instance default {
        type default
        protocols {
            bgp {
                bgp-label {
                    bgp-vpn {
                        dynamic-label-block range-6-bgp-lu
                    }
                }
```

The label block is shared by the EVPN inter-AS model B and EVPN NHS-RR features.

Note:
The **inter-as-vpn true** command has the same function as the **keep-all-routes** command for keeping the routes in the RIB.

The following shows the configuration of the default network instance of border router br4 in Figure 51: EVPN and IP-VPN services on inter-AS option B:

```
--{ + candidate shared default }--[  ]--
A:br4# info with-context network-instance default
    network-instance default {
        type default
        interface ethernet-1/10.0 {
        }
        interface ethernet-1/11.0 {
        }
        interface ethernet-1/12.0 {
        }
        interface system0.0 {
        }
```


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

```
        protocols {
            bgp {
                autonomous-system 65001
                router-id 10.0.0.4
                bgp-label {
                    bgp-vpn {
                        dynamic-label-block range-6-bgp-lu
                    }
                    bgp-ipvpn {
                        next-hop-resolution {
                            ipv4-next-hops {
                                tunnel-resolution {
                                    allowed-tunnel-types [
                                        ldp
                                        sr-isis
                                    ]
                                }
                            }
                        }
                    }
                }
                ebgp-default-policy {
                    import-reject-all false
                    export-reject-all false
                }
                afi-safi evpn {
                    admin-state enable
                    evpn {
                        inter-as-vpn true
                        rapid-update true
                        default-received-encapsulation mpls
                        next-hop-resolution {
                            ipv4-next-hops {
                                tunnel-resolution {
                                    allowed-tunnel-types [
                                        bgp
                                        ldp
                                        sr-isis
                                    ]
                                }
                            }
                        }
                    }
                }
                afi-safi l3vpn-ipv4-unicast {
                    admin-state enable
                    l3vpn-ipv4-unicast {
                        inter-as-vpn true
                        rapid-update true
                    }
                }
                group overlay-ebgp {
                    peer-as 65023
                    afi-safi evpn {
                    }
                    timers {
                        connect-retry 1
                        minimum-advertisement-interval 1
                    }
                    trace-options {
                        flag update {
                            modifier detail
                        }
                    }
```


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

```
            }
            group overlay-ibgp {
                peer-as 65001
                afi-safi evpn {
                }
                timers {
                    connect-retry 1
                    minimum-advertisement-interval 1
                }
                trace-options {
                    flag update {
                        modifier detail
                    }
                }
            }
            neighbor 10.4.5.2 {
                peer-group overlay-ebgp
            }
            neighbor 10.4.6.2 {
                peer-group overlay-ebgp
            }
            neighbor 10.0.0.1 {
                peer-group overlay-ibgp
            }
        }
        ldp {
            admin-state enable
            dynamic-label-block range-1-ldp
            discovery {
                interfaces {
                    interface ethernet-1/10.0 {
                        ipv4 {
                            admin-state enable
                        }
                    }
                    interface ethernet-1/11.0 {
                        ipv4 {
                            admin-state enable
                        }
                    }
                    interface ethernet-1/12.0 {
                        ipv4 {
                            admin-state enable
                        }
                    }
                }
            }
        }
        isis {
            dynamic-label-block range-3-srgb
            instance i14 {
                admin-state enable
                instance-id 1
                level-capability L2
                iid-tlv true
                net [
                    49.0001.0000.0000.0004.00
                ]
                segment-routing {
                    mpls {
                        dynamic-adjacency-sids {
                            all-interfaces true
                        }
                    }
```


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

```
                    }
                    interface ethernet-1/10.0 {
                        circuit-type point-to-point
                        ipv4-unicast {
                            admin-state enable
                        }
                    }
                    interface system0.0 {
                        passive true
                        ipv4-unicast {
                            admin-state enable
                        }
                    }
                }
            }
        }
        segment-routing {
            mpls {
                global-block {
                    label-range range-2-srgb
                }
                local-prefix-sid 1 {
                    interface system0.0
                    ipv4-label-index 4
                }
            }
        }
    }
```

The following shows the configuration of border router br4's label allocation:

```
--{ + candidate shared default }--[  ]--
A:br4# info with-context system mpls
    system {
        mpls {
            label-ranges {
                static range-2-srgb {
                    shared true
                    start-label 100001
                    end-label 120000
                }
                static range-5-static-services {
                    shared false
                    start-label 3000
                    end-label 4000
                }
                dynamic range-1-ldp {
                    start-label 100
                    end-label 200
                }
                dynamic range-3-srgb {
                    start-label 120001
                    end-label 120999
                }
                dynamic range-4-evpn {
                    start-label 500
                    end-label 699
                }
                dynamic range-5-services {
                    start-label 1000
                    end-label 2000
                }
                dynamic range-6-bgp-lu {
```


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

```
                    start-label 122001
                    end-label 122201
                }
            }
            services {
                evpn {
                    dynamic-label-block range-4-evpn
                }
                network-instance {
                    dynamic-label-block range-5-services
                }
            }
        }
    }
```

The **next-hop-self-route-reflector** command requires the configuration of a border router as a RR. The behavior of this command is equivalent to the **inter-as-vpn** command, with one difference: the use of **next-hop-self-route-reflector** allows the border router to receive and readvertise routes to RR clients within the same AS.

BGP next-hop resolution for EVPN/IP-VPN routes

NHS-RR and inter-AS option B routes all use the **next-hop-resolution** grouping. This grouping is only used when the following options are enabled:

- **inter-as-vpn**
- **nhsrr-evpn**

The **next-hop-resolution** command allows the disabling of route resolution. By default, routes are not ignored. If a longest prefix match (LPM) FIB lookup provides a route that is not local or static, and there is no tunnel, the route is unresolved. This command also allows the:

- resolution of tunnels through a fallback to FIB lookup
- resolution of local and static routes to the next hop in the absence of resolving tunnels

BGP receives next-hop resolution information from the following contexts:

- For IP-VPN:

```
--{ * candidate shared default }--[  ]--
A:srl1# info with-context network-instance default
    network-instance default {
        protocols {
            bgp {
                bgp-label {
                    bgp-ipvpn {
                        next-hop-resolution {
                                }
                            }
                        }
                    }
                }
```

- For EVPN:

```
--{ * candidate shared default }--[  ]--
A:srl3# info with-context network-instance default
    network-instance default {
        protocols {
            bgp {
```


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

```
                afi-safi evpn {
                    next-hop-resolution {
                    }
                }
            }
        }
    }
```

The **next-hop-resolution** configuration on the ASBR on the default network instance affects EVPN-MPLS routes but not EVPN-VXLAN routes. EVPN-VXLAN routes also ignore everything under the **next-hop-resolution** context.

The **next-hop-resolution tunnel-resolution allowed-tunnel-types** leaf restricts VXLAN and only allows MPLS tunnels.

The following describes the EVPN next-hop resolution logic:

1. If the EVPN route type is an ES route (route type 4) then the route is resolved to any route in the default network instance route table, regardless of encapsulation type or node type.
2. If the EVPN route type is different from an ES route, the following logic is followed:
   a. If the route has an encapsulation type of VXLAN or if there is no encapsulation type found and **default-received-encapsulation** is set to VXLAN then the route is resolved over any RTM route.
   b. If the route has an encapsulation type of MPLS or **default-received-encapsulation** is set to MPLS then the resolution is selected based on the default network instance ASBR or service configuration.

6.2.6.1 Displaying next-hop-self route reflector and inter-AS option B information

Procedure

You can display the service Incoming Label Mapping (ILM) information and the next hop resolution using the **info from state** command.

Example: Swapped service labels

The following example displays the swapped labels using the **info from state** command. The BGP-RIB provides the state information for each imported and exported route.

```
// example for EVPN IFL route on a model B ASBR:

--{ candidate shared default }--[  ]--
A:br4# info from state with-context network-instance default bgp-rib evpn rib-in-out rib-in-post ip-
prefix-routes 10.0.0.2:3 ethernet-tag-id 0 ip-prefix-length 24 ip-prefix 10.20.20.
0/24 neighbor 10.4.5.2
    network-instance default {
        bgp-rib {
            evpn {
                rib-in-out {
                    rib-in-post {
                        ip-prefix-routes 10.0.0.2:3 ethernet-tag-id 0 ip-prefix-length 24 ip-prefix
 10.20.20.0/24 neighbor 10.4.5.2 {
                            esi 00:00:00:00:00:00:00:00:00:00
                            gateway-ip 0.0.0.0
                            attr-id 125
                            last-modified "2024-04-10T12:46:50.200Z (2 hours ago)"
                            used-route false
                            valid-route true
                            best-route true
```


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

```
                            stale-route false
                            pending-delete false
                            tie-break-reason none
                            label {
                                value 122010  // received service label
                                value-type mpls-label
                            }
                            invalid-reason {
                                rejected-route false
                                as-loop false
                                next-hop-unresolved false
                                cluster-loop false
                                label-allocation-failed false
                                fib-programming-failed false
                            }
                        }
                    }
                }
            }
        }
    }
--{ candidate shared default }--[  ]--
A:br4# info from state with-context network-instance default bgp-rib evpn rib-in-out rib-out-post ip-
prefix-routes 10.0.0.2:3 ethernet-tag-id 0 ip-prefix-length 24 ip-prefix 10.20.20
.0/24 neighbor 10.0.0.1
    network-instance default {
        bgp-rib {
            evpn {
                rib-in-out {
                    rib-out-post {
                        ip-prefix-routes 10.0.0.2:3 ethernet-tag-id 0 ip-prefix-length 24 ip-prefix
 10.20.20.0/24 neighbor 10.0.0.1 {
                            esi 00:00:00:00:00:00:00:00:00:00
                            gateway-ip 0.0.0.0
                            attr-id 147
                            next-hop 10.0.0.4
                            label {
                                value 122007 // advertised label, 122010 is swapped to 122007
                                value-type mpls-label
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Next hop state in VRF for Inter-AS Option B

The swapped label value is used to find the next hop resolution.

```
// The service label entry 122007 provides the NHG and next hop

--{ candidate shared default }--[  ]--
A:br4# info from state with-context network-instance default route-table mpls label-entry 122007
    network-instance default {
        route-table {
            mpls {
                label-entry 122007 {
                    operation swap
                    entry-type bgp
                    last-app-update "2024-04-10T12:46:50.185Z (2 hours ago)"
                    next-hop-group 373030816119
```


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

```
                }
            }
        }
    }

--{ candidate shared default }--[  ]--
A:br4# info from state with-context network-instance default route-table next-hop-group 373030816119
    network-instance default {
        route-table {
            next-hop-group 373030816119 {
                backup-next-hop-group 0
                fib-programming {
                    last-successful-operation-type add
                    last-successful-operation-timestamp 2024-04-10T11:57:16.328Z
                    pending-operation-type none
                    last-failed-operation-type none
                }
                next-hop 0 {
                    next-hop 373030816115
                    resolved true
                }
            }
        }
    }

// In this example, the next hop is resolved to a local route 10.4.5.0 (ASBRs using single hop eBGP
 session)

--{ candidate shared default }--[  ]--
A:br4# info from state with-context network-instance default route-table next-hop 373030816115
    network-instance default {
        route-table {
            next-hop 373030816115 {
                type indirect
                ip-address 10.4.5.2
                resolving-route {
                    ip-prefix 10.4.5.0/30
                    route-type local
                    route-owner net_inst_mgr
                }
                mpls {
                    pushed-mpls-label-stack [
                        122010
                    ]
                }
            }
        }
    }
```

Example: MPLS route table

The following example displays the the MPLS route table which summarizes the swap operation, incoming label, outgoing label, and associated next hop.

```
--{ candidate shared default }--[  ]--
A:br4# show network-instance default route-table mpls
+--------+-----------+---------+-----------+----------------------+-----------------+---------------+
| Label  | Operation | Type    | Next      | Next-hop IP          | Next-hop        | Next-hop MPLS |
|        |           |         | Net-Inst  | (Type)               | Subinterface    | labels        |
+========+===========+=========+===========+======================+=================+===============+
| 100    | POP       | ldp     | default   |                      |                 |               |
| 100002 | SWAP      | sr-mpls | N/A       | 10.1.4.1 (mpls)      | ethernet-1/10.0 | 100002        |
| 100005 | POP       | sr-mpls | default   |                      |                 |               |
```


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

```
| 120002 | SWAP      | sr-mpls | N/A       | 10.1.4.1 (mpls)      | ethernet-1/10.0 | IMPLICIT_NULL |
| 122001 | SWAP      | bgp     | N/A       | 10.0.0.1 (indirect)  |                 | 1002          |
| 122002 | SWAP      | bgp     | N/A       | 10.0.0.1 (indirect   |                 | 1001          |
| 122003 | SWAP      | bgp     | N/A       | 10.0.0.1 (indirect)  |                 | 1003          |
| 122004 | SWAP      | bgp     | N/A       | 10.4.5.2 (indirect)  |                 | 122008        |
| 122005 | SWAP      | bgp     | N/A       | 10.4.5.2 (indirect)  |                 | 122011        |
| 122006 | SWAP      | bgp     | N/A       | 10.4.5.2 (indirect)  |                 | 122007        |
| 122007 | SWAP      | bgp     | N/A       | 10.4.5.2 (indirect)  |                 | 122010        |
| 122008 | SWAP      | bgp     | N/A       | 10.4.5.2 (indirect)  |                 | 122014        |
| 122009 | SWAP      | bgp     | N/A       | 10.4.5.2 (indirect)  |                 | 122009        |
| 122010 | SWAP      | bgp     | N/A       | 10.4.5.2 (indirect)  |                 | 122012        |
| 122011 | SWAP      | bgp     | N/A       | 10.4.5.2 (indirect)  |                 | 122013        |
+--------+-----------+---------+-----------+----------------------+-----------------+---------------+
// 122007 swapped with 122010 with next hop 10.4.5.2
```

6.3 Combined ECMP for BGP PE-CE and EVPN IFL/IP-VPN

Note: This feature is supported on 7220 IXR-D*x*, 7250 IXR Gen 2, 7250 IXR Gen 2c+, and 7250 IXR Gen 3 platforms.

SR Linux supports the combination of EVPN IFL, IP-VPN, and BGP PE-CE routes in a single ECMP set or next-hop group. The combination of routes is also supported in unequal ECMP.

Combined ECMP is supported along with IP Aliasing, unequal ECMP and resilient hashing for EVPN IFL routes.

Combined ECMP is represented in a route table as a single route with a combined next-hop group. When at least one of the next hops is coming from a BGP PE-CE route, the resulting route type is BGP.

Table 8: All possible route combinations for combined BGP PE-CE weighted ECMP shows all the possible route combinations and the resulting route types. The resulting route follows the best path selection result across the BGP owners of the route.

Table 8: All possible route combinations for combined BGP PE-CE weighted ECMP

Route combination	Resulting route type
• `bgp` • `bgp-ipvpn`	BGP
• `bgp` • `bgp-evpn`	BGP
• `bgp-evpn` • `bgp-ipvpn`	BGP-EVPN
• `bgp` • `bgp-evpn` • `bgp-ipvpn`	BGP

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

The range of the number of ECMP paths in a combined set is the minimum of the configured ECMP paths of the contributing routes. The individual ECMP path ranges are:

- `bgp-evpn.ecmp[=1..64]`
- `bgp-ipvpn.ecmp[=1..64]`
- `bgp.multipath.max-paths-level-1[=1..64]`

To show the maximum ECMP number of paths, the following is an example configuration:

- `bgp-evpn.ecmp[=10]`
- `bgp-ipvpn.ecmp[=12]`
- `bgp.multipath.max-paths-level-1[=15]`

In this example, the following are true:

- If at least one `bgp-evpn` is present in the combined route, the maximum number of ECMP paths used is 10.
- Else, if at least one `bgp-ipvpn` route is present in the combined route, the maximum number of ECMP paths used is 12.
- Else, the maximum number of ECMP paths is 15.

The combination of multiple BGP PE-CE routes in a single ECMP set is configured using the **combined-ecmp** command.

```
--{ * candidate shared default }--[  ]--
A:srl1# info with-context network-instance ip-vrf
    network-instance ip-vrf {
        protocols {
            bgp-vpn {
                combined-ecmp {
                }
            }
        }
    }
```

Candidate routes to be combined are compared against the BGP best path selection tie-breakers and routes can only be combined if all the best path selection criteria up to the MED comparison are equal. That is, two routes of different owners can be combined for ECMP purposes only if the following attributes match on both routes:

1. Valid routes
2. Origin validation state
3. Route table PREFERENCE
4. BGP local preference
5. BGP AS_PATH
6. BGP origin
7. BGP MED

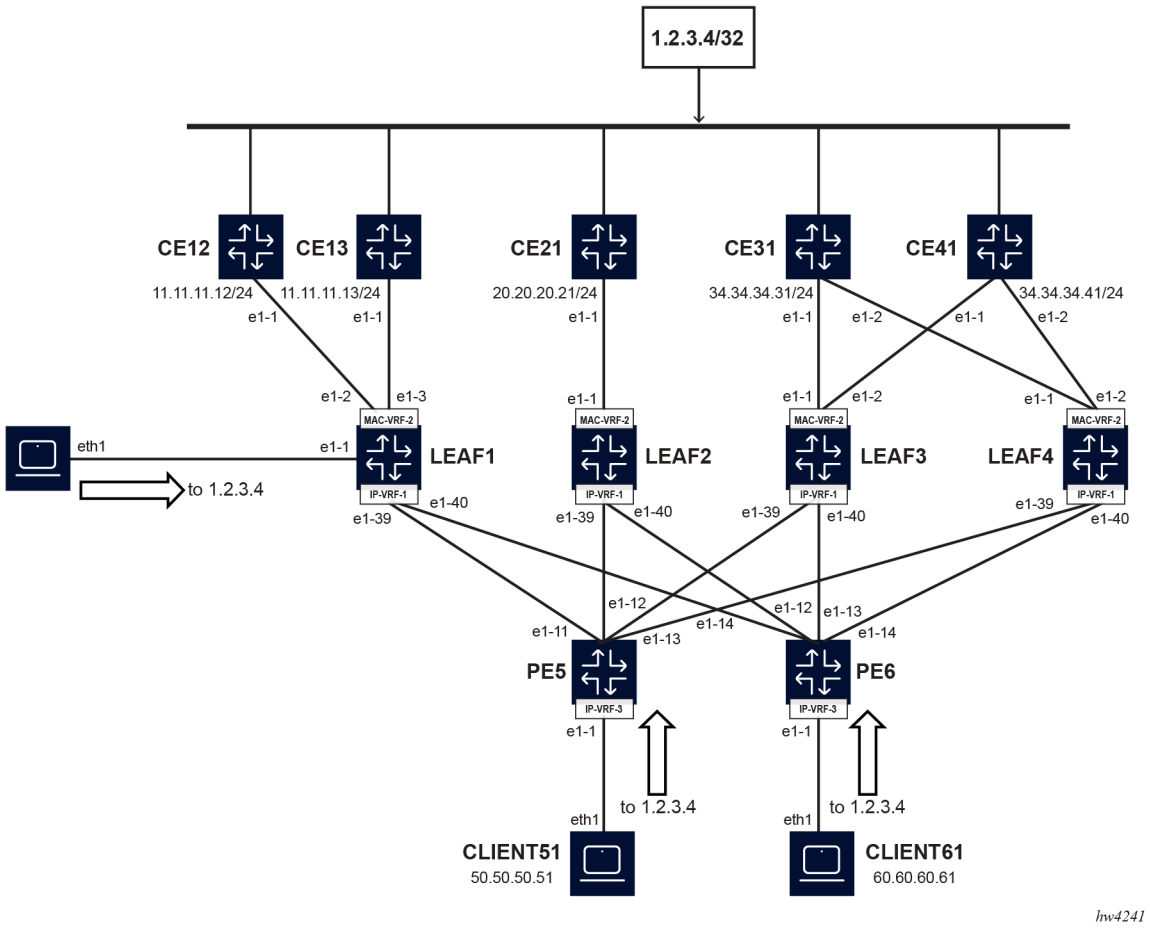
When multiple routes of the same or different owner for the same prefix are received, the selection is affected by the configuration of **combined-ecmp**, as indicated in Best path selection for EVPN IFL routes.

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

6.3.1 Combined ECMP example

The following figure shows an example of combined ECMP:

Figure 52: Combined ECMP



The figure shows that CE12, CE13, CE21, CE31 and CE41 are all connected to the host 1.2.3.4/32 and advertise this in BGP PE-CE to their connected leaf router. Leaf1 receives the 1.2.3.4/32 prefix via BGP and BGP-EVPN. By configuring **combined-ecmp**, leaf1 can install a single BGP route to 1.2.3.4/32 with a combined next-hop group that includes the PE-CE routes and remote next hops from EVPN IFL routes, with or without weights and with or without ESIs.

The following shows the configuration of leaf1:

```
--{ * candidate shared default }--[  ]--
A:srl1# info with-context network-instance IP-VRF-1
    network-instance IP-VRF-1 {
        type ip-vrf
        description "IP-VRF-2 EVPN-MPLS uecmp"
        ip-load-balancing {
            resilient-hash-prefix 1.2.0.0/16 {
```


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

```
                hash-buckets-per-path 6
                max-paths 10
            }
        }
        interface ethernet-1/1.1 {
        }
        interface irb0.1 {
        }
        interface lo0.1 {
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    encapsulation-type mpls
                    evi 1
                    ecmp 64
                    mpls {
                        next-hop-resolution {
                            allowed-tunnel-types [
                                ldp
                                sr-isis
                            ]
                        }
                    }
                    routes {
                        route-table {
                            ip-prefix {
                                evpn-link-bandwidth {
                                    advertise {
                                        weight dynamic
                                    }
                                    weighted-ecmp {
                                        admin-state enable
                                        max-ecmp-hash-buckets-per-next-hop-group 64
                                    }
                                }
                            }
                        }
                    }
                }
            }
            bgp {
                admin-state enable
                autonomous-system 64500
                router-id 1.1.1.1
                ebgp-default-policy {
                    import-reject-all false
                    export-reject-all false
                }
                afi-safi ipv4-unicast {
                    admin-state enable
                    multipath {
                        maximum-paths 64
                    }
                }
                trace-options {
                    flag packets {
                        modifier detail
                    }
                    flag update {
                        modifier detail
                    }
                }
                group pe-ce {
```


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

```
                    multihop {
                        admin-state enable
                        maximum-hops 10
                    }
                    afi-safi ipv4-unicast {
                        ipv4-unicast {
                            link-bandwidth {
                                add-next-hop-count-to-received-bgp-routes 1
                            }
                        }
                    }
                    timers {
                        connect-retry 1
                        minimum-advertisement-interval 1
                    }
                    trace-options {
                        flag update {
                            modifier detail
                        }
                    }
                    transport {
                        local-address 1.1.1.1
                    }
                }
                neighbor 11.11.11.12 {
                    peer-as 64512
                    peer-group pe-ce
                }
                neighbor 11.11.11.13 {
                    peer-as 64513
                    peer-group pe-ce
                }
            }
            bgp-vpn {
                combined-ecmp {
                }
                bgp-instance 1 {
                }
            }
        }
    }
```

The route table for prefix 1.2.3.4/32 is displayed as follows:

```
--{ + candidate shared default }--[ network-instance IP-VRF-1 ]--
A:leaf1# show route-table ipv4-unicast prefix 1.2.3.4/32
-------------------------------------------------------------------------------------------------------
IPv4 unicast route table of network instance IP-VRF-1
-------------------------------------------------------------------------------------------------------
+------------+----+-------+---------+--------+----------+--------+-----+-----------------+-----------+
| Prefix     | ID | Route | Route   | Active | Origin   | Metric | Pref| Next-hop        | Next-hop  |
|            |    | Type  | Owner   |        | Network  |        |     | (Type)          | Interface |
|            |    |       |         |        | Instance |        |     |                 |           |
+============+====+=======+=========+========+==========+========+=====+=================+===========+
| 1.2.3.4/32 | 0  | bgp   | bgp_mgr | True   | IP-VRF-1 | 0      | 170 | 11.11.11.0/24   | irb0.1    |
|            |    |       |         |        |          |        |     | (indirect/local)| irb0.1    |
|            |    |       |         |        |          |        |     | 11.11.11.0/24   |           |
|            |    |       |         |        |          |        |     | (indirect/local)|           |
|            |    |       |         |        |          |        |     | 100.0.0.2/32    |           |
|            |    |       |         |        |          |        |     | (indirect/ldp)  |           |
|            |    |       |         |        |          |        |     | 100.0.0.3/32    |           |
|            |    |       |         |        |          |        |     | (indirect/ldp)  |           |
|            |    |       |         |        |          |        |     | 100.0.0.4/32    |           |
```


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

```
|            |    |       |         |        |          |        |     | (indirect/ldp)  |           |
+------------+----+-------+---------+--------+----------+--------+-----+-----------------+-----------+
-------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------
```

The following shows the route table state for the route:

```
--{ + candidate shared default }--[ network-instance IP-VRF-1 ]--
A:leaf1# info from state with-context route-table ipv4-unicast route 1.2.3.4/32 id 0 route-type
 bgp route-owner bgp_mgr origin-network-instance IP-VRF-1
leakable false
    metric 0
    preference 170
    active true
    last-app-update "2024-07-02T09:06:02.838Z (an hour ago)"
    next-hop-group 347400906964
    next-hop-group-network-instance IP-VRF-1
    resilient-hash true
    fib-programming {
        suppressed false
        last-successful-operation-type modify
        last-successful-operation-timestamp "2024-07-02T09:06:02.839Z (an hour ago)"
        pending-operation-type none
        last-failed-operation-type none
    }
--{ + candidate shared default }--[ network-instance IP-VRF-1 ]--
A:leaf1# info from state with-context route-table next-hop-group 347400906964
    backup-next-hop-group 0
    fib-programming {
        last-successful-operation-type modify
        last-successful-operation-timestamp "2024-07-02T09:06:02.839Z (an hour ago)"
        pending-operation-type none
        last-failed-operation-type none
    }
    next-hop 0 {
        next-hop 347400907006
        resolved true
    }
    next-hop 1 {
        next-hop 347400907007
        resolved true
    }
    next-hop 2 {
        next-hop 347400906982
        resolved true
    }
    next-hop 3 {
        next-hop 347400906999
        resolved true
    }
    next-hop 4 {
        next-hop 347400906996
    resolved true
    }
--{ + candidate shared default }--[ network-instance IP-VRF-1 ]--
A:leaf1# info from state with-context route-table next-hop
 {347400907006,347400907007,347400906982,347400906999,347400906996}
    next-hop 347400907006 {
        type indirect
        ip-address 11.11.11.12
            resolving-route {
            ip-prefix 11.11.11.0/24
            route-type local
```


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

```
            route-owner net_inst_mgr
            }
        }
    next-hop 347400907007 {
        type indirect
        ip-address 11.11.11.13
            resolving-route {
            ip-prefix 11.11.11.0/24
            route-type local
            route-owner net_inst_mgr
            }
        }
    next-hop 347400906982 {
        type indirect
        ip-address 100.0.0.2
        resolving-tunnel {
            ip-prefix 100.0.0.2/32
            tunnel-type ldp
            tunnel-owner ldp_mgr
        }
    mpls {
        pushed-mpls-label-stack [
            1000
            ]
        }
    }
    next-hop 347400906999 {
        type indirect
        ip-address 100.0.0.3
        resolving-tunnel {
            ip-prefix 100.0.0.3/32
            tunnel-type ldp
            tunnel-owner ldp_mgr
        }
    mpls {
        pushed-mpls-label-stack [
            1002
            ]
        }
    }
    next-hop 347400906996 {
        type indirect
        ip-address 100.0.0.4
        resolving-tunnel {
            ip-prefix 100.0.0.4/32
            tunnel-type ldp
            tunnel-owner ldp_mgr
        }
    mpls {
        pushed-mpls-label-stack [
            1001
            ]
        }
    }
```

6.4 Micro-segmentation

Micro-segmentation is a security capability for data centers and enterprises intended to prevent lateral movement of security threats within the network. It divides networks into smaller, isolated zones (known as micro-segments) and establishes rules to restrict traffic between them. An example of a micro-segment

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

defined within an enterprise network is a set of homogeneous devices such as DNS servers, printers, print servers, client applications, or server applications.

Micro-segmentation uses group-based policies (GBPs) to control access between segments. A GBP defines a set of groups, with a tag value assigned to each group. The GBP associates the groups with a subinterface or IP prefix and specifies an ACL to enforce access rules based on group tagging.

Controlling access using GBPs provides a more efficient use of system TCAM resources than using IP addresses. For example, an interface ACL configuration to control traffic between 50 printers and 3 print servers may require a minimum of 150 TCAM entries if the ACL uses source and destination IP addresses as match criteria. In a GBP, the printers and print servers are placed into separate groups, and the GBP maps their IP addresses to group values. An ACL configured within the GBP filters traffic based on the group values, so that traffic from the printers group to the print server group is accepted, and all other traffic from the printers group is dropped. Filtering based on source and destination group in this way requires only a single TCAM entry to accept traffic from the printers group to the print servers group, and another TCAM entry to drop all other traffic from the printers group.

Associating group-based policies with network resources

Group-based policies can be associated with network resources in the following ways:

- A GBP configured in a network-instance can include any or all of the subinterfaces within the network-instance, with a group assigned to each subinterface. See Associating groups with interfaces.
- SR Linux can use information in BGP routes to derive the group tag to associate with an IP host or IP prefix. The group tag from the Group Policy ID extended community can be included in routes installed in the FIB, and the FIB-programmed group tag can be copied into the Group Policy ID extended community in routes advertised to BGP peers. See GBP tagging for BGP routes.
- A GBP group can be associated with a static route, and a static route associated with a GBP group can be configured to follow the forwarding behavior of a parent route already present in the route table of the same network-instance. See Static routes and micro-segmentation.
- A set of group tags can be carried in the Group Policy ID extended community, and routing policies can use the Group Policy ID extended community as a match condition for BGP routes. See Routing policies and micro-segmentation.

Platform support

Micro-segmentation is supported on the following platforms:

- 7220 IXR-D2/D3
- 7220 IXR-D4/D5

On the 7220 IXR-D4, Layer 2 micro-segmentation is supported with the existing datapath resource management layout. Layer 3 micro-segmentation on the 7220 IXR-D4 requires the **td4-mode** parameter within the context **platform resource-management unified-forwarding-resources** to be set to 2 and XDP to be restarted.

On the 7220 IXR-D5, Layer 3 micro-segmentation is not supported.

For information about configuring **td4-mode**, see the "Changing datapath resource management settings" section in the *SR Linux Configuration Basics Guide*.

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

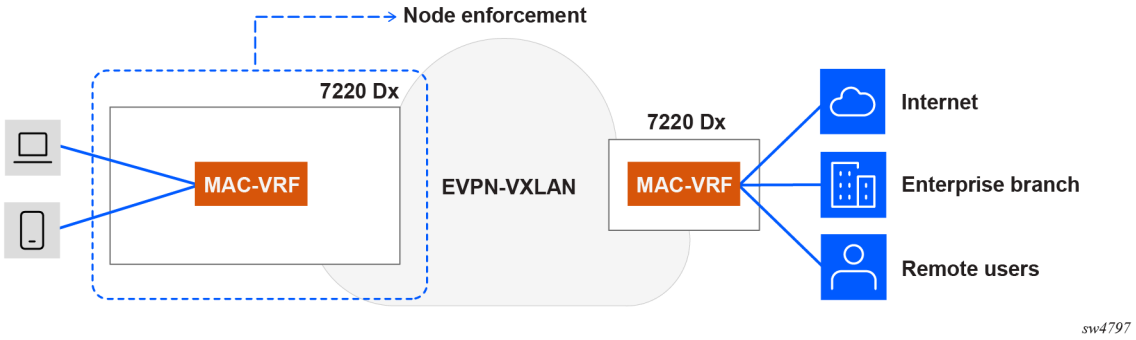
6.4.1 Micro-segmentation topologies

Micro-segmentation can be used in the following topology models. For more information about these models and configuration examples, see Group-based policies in VPN services.

Layer 2 only model

The Layer 2 model uses MAC-VRFs only. The following illustration shows a Layer 2 only model for micro-segmentation.

Figure 53: Layer 2 model for micro-segmentation

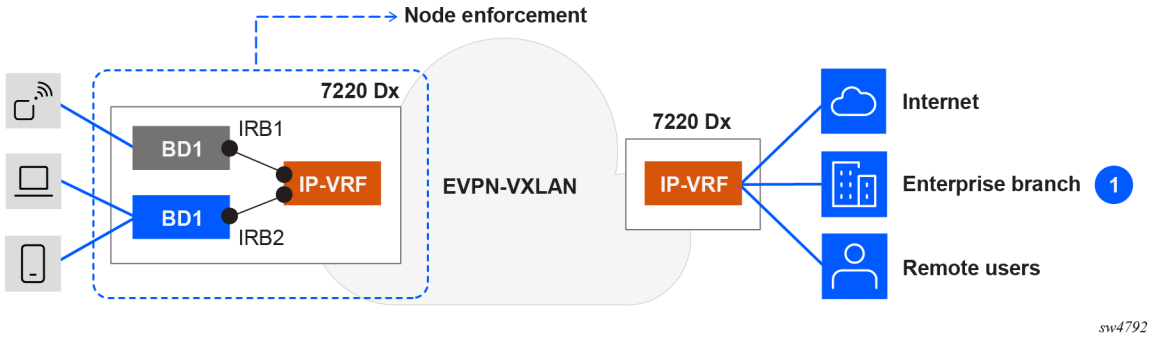


In the group-based policy for each MAC-VRF, groups are associated with Layer 2 subinterfaces. GBP tags are distributed in EVPN routes. GBP ACLs are configured so that the ingress node enforces intra-VRF communication. Source and destination GBP tags are found in MAC lookup.

Distributed IRB model

The distributed IRB model enforces policies at the ingress node, which allows unwanted traffic to be dropped at the point of entry and avoids unnecessary bandwidth consumption within the network. The following illustration shows a distributed IRB model for micro-segmentation.

Figure 54: Distributed IRB model for micro-segmentation



In the distributed IRB model, the ingress node connects the broadcast domains via IRBs into an IP-VRF. Groups are associated with Layer 2 subinterfaces and static routes. GBP tags are distributed in EVPN routes. GBP ACLs are configured so that the ingress node enforces intra-subnet and inter-subnet communication.

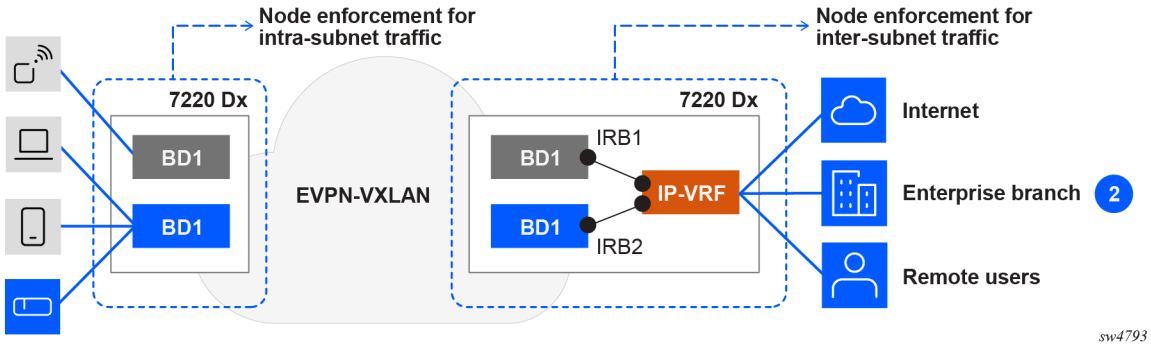
© 2026 Nokia.
Use subject to Terms available at: www.nokia.com/terms.

- For intra-subnet communication, the source and destination GBP tags are found using MAC address lookups in the MAC table, which stores GBP tag associations used for Layer 2 policy enforcement.
- For inter-subnet communication, the source GBP tags are found using MAC table lookups, and the destination GBP tags are found from an LPM (Longest Prefix Match) on the corresponding IP-VRF route table.

Centralized IRB model

In the centralized IRB model, access nodes are connected to broadcast domains, and inter-subnet communication is handled exclusively by a centralized IRB on a central node. The following illustration shows a centralized IRB model for micro-segmentation.

Figure 55: Centralized IRB model



In the centralized IRB model, the ingress node aggregates the MAC-VRF broadcast domains over an EVPN-VXLAN fabric. A central node connects the broadcast domains via IRBs into an IP-VRF. The access node forwards traffic based on MAC lookups, while inter-subnet forwarding occurs at the central node. Groups are associated with Layer 2 subinterfaces and static routes. GBP tags are distributed in EVPN routes. GBP ACLs are configured so that the ingress node enforces intra-subnet communication, and the egress node enforces inter-subnet communication on ingress of the IP-VRF IRB subinterfaces. Note that there is no GBP ACL on ingress of the MAC-VRF of the egress node.

- For intra-subnet communication, the source and destination GBP tags are obtained from MAC table lookups on the ingress node.
- For inter-subnet communication, the source and destination GBP tags are derived from either LPM lookups of the source and destination IP addresses in the route table of the egress node, or the source tag is from a source MAC lookup and destination tag from a LPM table lookup.

ARP and ND traffic cannot be snooped over VXLAN. As a result, the centralized IRB model requires Layer 2 proxy-ARP/ND on the access node. This ensures that ARP/ND packets are intercepted at the ingress node and advertised via EVPN, enabling correct resolution at the central node.

There is no GBP configuration in the MAC-VRFs on the egress node.

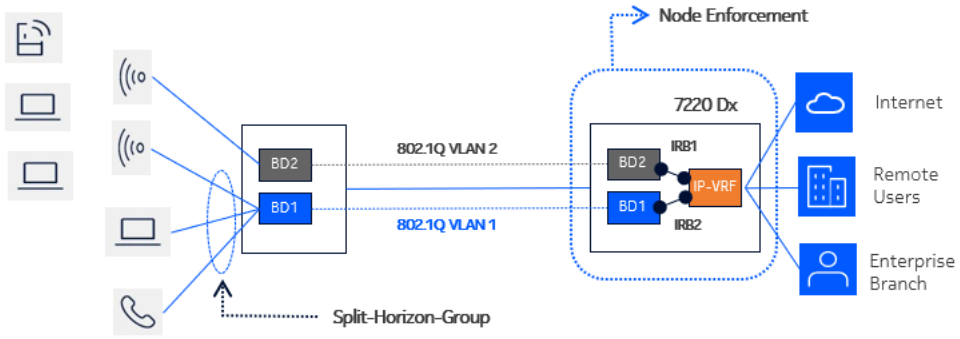
Centralized IRB model for Layer 2 aggregation

In a network with Layer 2 access nodes that lack GBP enforcement capabilities, policy enforcement for both intra- and inter-subnet communication is performed at the egress node. Intra-subnet communication relies on routing through a central IRB node, typically leveraging Layer 3 proxy-ARP/ND and host routes in the IP-VRF route table. In this architecture, enforcement for both intra- and inter-subnet traffic is based on GBP tags derived from LPM lookups of the source and destination IP addresses.

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

The following illustration shows a centralized IRB model for Layer 2 aggregation.

Figure 56: Centralized IRB model for Layer 2 aggregation



In this model, ingress split-horizon-groups are configured on the access node to avoid local intra-subnet communication. GBP tags are assigned to subinterfaces on the access node, as well as to subinterfaces and static routes on the egress node. Host routes (/32 routes) are populated in the IP-VRF route-table.

GBP ACLs are configured so that the egress node enforces inter-subnet and intra-subnet communication:

- For inter-subnet communication, the source GBP tags are found in LPM lookup, and the destination GBP tags are found in LPM lookup.
- For intra-subnet communication, the egress node relies on Layer 3 proxy-ARP/ND and host routing, specifically populating /32 host routes in the central node's route table. This approach prevents local switching of traffic within the ingress node's broadcast domain and allows full enforcement at the egress node.

6.4.2 Group-based policies

In a network-instance, a group-based policy (GBP) defines groups, their association rules, and the GBP ACL filter that applies to the groups. The following parameters are configured in a GBP:

- **group**

  The list of groups (micro-segments) used in this network-instance. See Creating a group-based policy
- **tag-value**

  The tag value assigned to a group; this value is used in BGP route advertisement and the GBP ACL. This value is mandatory.
- **interface**

  The list of subinterfaces with their associated GBP group. See Associating groups with interfaces.
- **acl**

  The GBP ACL definition for this network-instance. See the *SR Linux ACL and Traffic Steering Guide*.

In a GBP, you configure the complete list of groups and associated tag-values used in the network-instance, even if not all groups are locally connected to the switch. For example, a group may only exist on a remote switch and its route advertised over BGP. To display a remote route group name in info from state or show commands, this group must be locally configured; otherwise, only the tag-value found in the BGP

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

extended-community is displayed as associated with the route. The group name is also mandatory for use in ACL source or destination match criteria.

A group-based-policy can be configured only in network-instances of type IP-VRF and MAC-VRF, and not type default.

Valid tag values for IP-VRFs and MAC-VRFs

Tag values assigned to a group are unique per group within the network-instance. The valid tag value range depends on the platform and VRF type, as shown in the following tables:

Table 9: Valid tag value range for IP-VRF on 7220 IXR platforms

	7220 IXR-D2/D3	7220 IXR-D4/D5
Tag field length	7 bits	14 bits
Value range	0 to 127	0 to 16383

Table 10: Valid tag value range for MAC-VRF on 7220 IXR platforms

	7220 IXR-D1/D2/D3	7220 IXR-D4/D5
Tag field length	8 bits	14 bits
Value range	0 to 255	0 to 16383

GBP ACLs always use tag values in this per-platform range for MAC and IP-VRF ACL TCAM entries because they use the configured GBP tag value for the group.

For TCAM allocation, the system handles Layer 2 GBP tags and Layer 3 GBP tags differently. For GBP ACLs configured in a MAC-VRF, the system handles only Layer 2 GBP tags, so it configures a single TCAM entry for each GBP ACL entry with `L2srcTAG` and `L3dstTAG`.

For GBP ACLs configured in an IP-VRF:

- The destination tag is always looked up in the LPM table (Layer 3 GBP tag) as `L3dstTAG`.
- The source tag may come from the MAC table (Layer 2 GBP tag) in the case of a distributed design with a MAC-VRF and IP-VRF on the same system and IRB interfaces, or the source tag may come from the source IP lookup (Layer 3 GBP tag).
- Two TCAM sub-entries are programmed with `L2srcTAG` and `L3srcTAG` for each ACL entry that uses a `source-group` match condition, provided that the GBP tag-value in the IP-VRF is within the allowed range for of MAC-VRF GBP tag values for the platform.
- If the tag value in the IP-VRF is larger than the supported maximum tag value for a MAC-VRF, one TCAM entry is programmed with `L3srcTAG` alone.

The same tag value can be reused across multiple VRFs and does not need to be unique network-wide, with the following exception: for Layer 3 micro-segmentation with route leaking, the tag value for the leaked groups and routes must be consistent between the VRFs involved.

Tag value 0 is assigned by default to LPM and MAC table entries that do not have a tag value assigned through configuration or by BGP. You can configure a group name such as "untagged" with tag value 0 that can be used in the GBP ACL.

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

Creating a group-based policy

To configure groups for a group-based policy in a network-instance, specify a name and tag value for each group. In the following example, three groups are created, and a tag value is assigned to each one.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance gbpex group-based-policy
    network-instance gbpex {
        group-based-policy {
            group grp1 {
                tag-value 100
            }
            group grp2 {
                tag-value 101
            }
            group grp3 {
                tag-value 103
            }
        }
    }
```

6.4.3 Associating groups with interfaces

The GBP configured in a network-instance can include any or all of the subinterfaces within the network-instance. The GBP can assign a group to each subinterface.

When the GBP is enforced on the ingress node:

- For routed and IRB subinterfaces: the source tag is assigned to the IP prefixes configured on this subinterface. These prefixes can be advertised in BGP EVPN routes with their associated GPID.
- For bridged subinterfaces: the source tag is assigned to the MAC address ingressing on this subinterface.

When the GBP is enforced on the egress node:

- For routed and IRB subinterfaces: the destination tag is assigned to the IP packets egressing on this subinterface.
- For bridged subinterfaces: The source tag is assigned to the Layer 2 packets egressing on this subinterface.

To assign a group to a subinterface, add the subinterface to the GBP and specify the group to which it belongs. The following example configures a GBP in an IP-VRF network-instance. In the GBP, a group is assigned to each subinterface.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance gbpex
    network-instance gbpex {
        type ip-vrf
        interface ethernet-1/1.1 {
        }
        interface ethernet-1/2.1 {
        }
        group-based-policy {
            group grp1 {
                tag-value 100
            }
            group grp2 {
                tag-value 101
```


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

```
            }
            interface ethernet-1/1.1 {
                group [
                    grp1
                ]
            }
            interface ethernet-1/2.1 {
                group [
                    grp2
                ]
            }
        }
    }
```

6.4.4 GBP tagging for BGP routes

SR Linux can use information in BGP routes to derive the group tag to associate with an IP host or IP prefix. The group tag is carried in the Group Policy ID extended community (type=0x03, subtype=17) attached to the BGP route, as described in draft-lrss-bess-evpn-group-policy. You can configure SR Linux to include the group tag from this extended community (EC) in routes it installs in the FIB, and you can configure SR Linux to copy the FIB-programmed group tag into the Group Policy ID EC in routes advertised to BGP peers.

Installing BGP routes into the FIB with GBP information

In a micro-segmentation topology, there are situations when the group tag a switch associates with an IP host, or set of IP hosts in a prefix range, should be derived from the BGP route that the switch received for that IP host or IP prefix. The BGP route could be an IPv4-unicast route, an IPv6-unicast route, or an EVPN type-2 or type-5 route that was originated by a remote switch closer to the IP hosts covered by the route. In these situations, the group associated with the IP prefix is carried in the Group Policy ID EC attached to the BGP route.

For GBP ACLs to decide whether to accept or drop a packet with a source or destination address matched by a BGP route with a Group Policy ID extended community, the BGP route must be installed into the FIB with a group tag copied from the extended community.

To configure SR Linux to do this, enable `group-based-policy.install-from-community` for the appropriate address family. For example, the following configuration applies to IPv4 unicast BGP routes installed into the FIB:

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance gbpex protocols bgp
    network-instance gbpex {
        protocols {
            bgp {
                afi-safi ipv4-unicast {
                    admin-state enable
                    ipv4-unicast {
                        group-based-policy {
                            install-from-community true
                        }
                    }
                }
            }
        }
    }
```


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

The BGP route may have been received on the wire without any Group Policy ID EC, but a VRF import policy or peer import policy added one. This is not handled any differently than receiving the EC on the wire.

The BGP route may carry multiple Group Policy ECs. In this case, SR Linux prefers the EC with zero in the Policy ID scope. If this applies to multiple ECs, the EC with the lowest Group Policy ID value of is chosen.

The Group Policy ID EC that BGP selects to install may have a non-zero value in the Policy ID Scope. This is not considered an error, and the Group Policy ID value is programmed as encoded.

The Group Policy ID EC that BGP selects to install may carry a Group Policy ID value that is out of range of the limits of the platform (see Valid tag values for IP-VRFs and MAC-VRFs). If the value is out of range, the BGP route is installed without any GBP tag.

Advertising BGP routes with Group Policy ID extended community

When a switch advertises a Group Policy ID EC in the BGP RIB-OUT, it may do so because it received the route from another device and is simply propagating it, or the switch added the EC by route policy action. It is also possible that the switch added the EC by copying the FIB-programmed GBP tag value into the Group Policy ID field of the EC. In this case, the FIB-programmed GBP tag value comes from the following:

- The group tag configured with `network-instance.group-based-policy.interface`
- Static route configuration
- Route policy action
- The BGP `install-from-community` operation

To configure SR Linux to copy the FIB-programmed GBP tag value into the Group Policy ID field of the EC, enable `group-based-policy.add-community` for the appropriate address family. The following example configures BGP to add the Group Policy ID EC to IPv4 unicast routes advertised to any BGP peer in this network-instance.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance gbpex protocols bgp
    network-instance gbpex {
        protocols {
            bgp {
                afi-safi ipv4-unicast {
                    admin-state enable
                    ipv4-unicast {
                        group-based-policy {
                            add-community true
                        }
                    }
                }
            }
        }
    }
```

6.4.5 Static routes and micro-segmentation

A GBP group can be associated with a static route, and a static route associated with a GBP group can be configured to follow the forwarding behavior of a parent route already present in the route table of the same network-instance.

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

Associating a GBP group with a static route

In a micro-segmentation configuration, you can assign the set of IP endpoints and hosts covered by a static route prefix to a GBP group. The association between the static route and GBP group occurs within the context of a single Layer 3 network-instance (default or a configured IP-VRF). For example, if a static route is installed into an IP-VRF, the static route can be associated with any the groups configured for the network-instance. However, it cannot be associated with groups defined under a different network-instance.

The following example specifies a group to be used with a static route. Each static route can be associated with only one group at a time.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance gbpex static-routes
    network-instance gbpex {
        static-routes {
            route 192.168.18.0/24 {
                group-based-policy {
                    group [
                        grp1
                    ]
                }
            }
        }
    }
```

Configuring a static route to follow another route

A static route can be configured to inherit the forwarding behavior of a parent route already present in the route table of the same network-instance. In this case, the NHG of the next-longest-prefix-match route becomes the effective NHG of the static route. This is useful in a micro-segmentation configuration where the enforcement switch may not have an IP route in its FIB that matches the intended boundary of a Layer 3 GBP group.

For example, consider a switch that has an IP interface with address 100.0.0.1/24 and has installed a route for 100.0.0.1/32 and a route for 100.0.0.0/24 in its FIB. In this subnet, hosts 100.0.0.4 and 100.0.0.5 comprise the group grp1. The switch does not have a route for 100.0.0.4/31 in its FIB that exactly covers just these two hosts, so a static route is configured for 100.0.0.4/31, and the static route is associated with group grp1. This static route is set to follow the next-longest-prefix-match route in the FIB, which causes the route to inherit the forwarding behavior of the 10.0.0.0/24 local route.

The following example configures static route 100.0.0.4/31, associates it with group grp1, and sets the `follow.next-longest` option to `true`, so that it follows the next-longest-prefix-match route when the static route itself is not present in the FIB.

```
--{ +* candidate shared default }--[  ]--
# info with-context network-instance gbpex static-routes
    network-instance gbpex {
        static-routes {
            route 100.0.0.4/31 {
                group-based-policy {
                    group [
                        grp1
                    ]
                }
                follow {
                    next-longest true
                }
            }
```


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

```
        }
    }
```

6.4.6 Routing policies and micro-segmentation

A set of group tags can be carried in the Group Policy ID extended community (type=0x03, subtype=17), and routing policies can use the Group Policy ID extended community as a match condition for BGP routes.

Configuring a Group Policy ID extended community

In a Group Policy ID extended community, a set of group tags is specified using the format `gbp-tag:x:y` to indicate member representation.

If the member format is `gbp-tag:<d1>:<d2>`, then <d1> is a decimal representation of the number encoded in the 16-bit Policy ID Scope field, and <d2> is a decimal representation of the number encoded in the 16-bit Group Policy ID field.

If the member format is `gbp-tag:<r>`, then the entire string is a regex matched against a string constructed from the extended community as follows: The sequence of characters `gbp-tag:` is followed by a decimal number representation of the Policy ID Scope, followed by another colon, followed by a decimal representation of the Group Policy ID.

The following terms are supported:

- single digit term – match if the character is x, where x can be a digit character [0-9] or a colon
- range term – match if the character is a digit with any value in the range `x..y` inclusive
- sub-expression – match if there is a sequence of characters matching the regular expression in brackets ()
- wildcard term (.) – match any single character, which can be a digit character [0-9] or a colon

The following example configures a Group Policy ID extended community that matches using a decimal representation of the 16-bit Policy ID Scope field and a decimal representation of 16-bit Group Policy ID field:

```
--{ + candidate shared default }--[  ]--
# info with-context routing-policy extended-community-set gbpecs
    routing-policy {
        extended-community-set gbpecs {
            member [
                gbp-tag:0:53
            ]
        }
    }
```

The following example configures a Group Policy ID extended community that uses a regular expression to match all Group Policy ID extended communities:

```
--{ + candidate shared default }--[  ]--
# info with-context routing-policy extended-community-set gbpecs
    routing-policy {
        extended-community-set gbpecs {
            member [
                "gbp-tag:.*"
            ]
```


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

```
        }
    }
```

See the *SR Linux Routing Protocols Guide* for additional examples of using `gbp-tag` in regular expressions.

Using a Group Policy ID extended community in a routing policy

The following example configures a routing policy that uses the Group Policy ID extended community as a match condition for BGP routes:

```
--{ +* candidate shared default }--[  ]--
# info routing-policy policy p1
    statement st1 {
        match {
            bgp {
                extended-community {
                    extended-community-set gbpecs
                    match-set-options all
                }
            }
        }
        action {
            policy-result accept
        }
    }
```

6.4.7 LPM source lookup mode

If an SR Linux device is deployed in a role that requires the source tag to be derived from the lookup of the source IP address in the LPM table, the device must be configured to operate in LPM source lookup mode. To do this, you configure the `platform.resource-management.group-based-policy.lpm-source-lookup` leaf to `true`. Each change in the value of this leaf requires a chassis reboot.

A change of `lpm-source-lookup` from `false` to `true` overrides the configured UFT profile on a 7220 IXR device and may reduce the overall IP route scale by 50% or more; a change of `lpm-source-lookup` from `true` to `false` reverts the UFT profile to standard configuration and restores IP route scale to expected levels.

When LPM source lookup mode is operationally active, the lookup applies to all IPv4 and IPv6 packets received on all routed subinterfaces of the default network-instance and all IP-VRF network-instances. For terminating VXLAN packets, the lookup is done post-decapsulation using the inner-header source IP address. If the LPM lookup for a source IP address finds a route without a tag, it is equivalent to tag 0.

Configuring LPM source lookup mode

To enable LPM source lookup mode, you set the **lpm-source-lookup** parameter to **true** and reboot the SR Linux device.

Example

The following example enables LPM source lookup mode:

```
--{ +* candidate shared default }--[  ]--
# info with-context platform resource-management group-based-policy
    platform {
        resource-management {
```


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

```
            group-based-policy {
                lpm-source-lookup true
            }
        }
    }
```

6.4.8 Group-based policies in VPN services

Note: Group-based policy services are supported on 7220 IXR-D2/D2L/D3/D3L/D4/D5/D5T platforms.

On the 7220 IXR-D4, Layer 2 group-based policy is supported with the existing datapath resource management layout. Layer 3 group-based policy on the 7220 IXR-D4 requires the **td4-mode** parameter within the context **platform resource-management unified-forwarding-resources** to be set to 2 and XDP to be restarted.

On the 7220 IXR-D5, Layer 3 group-based policy is not supported in this release.

For information about configuring **td4-mode**, see the "Changing datapath resource management settings" section in the *SR Linux Configuration Basics Guide*.

This chapter describes the components of group-based policies in VPN services on SR Linux.

Group-based policy (GBP) provides policy enforcement for traffic between hosts of the same tenant, where those hosts can be located in the same or different subnet. GBP configuration defines a set of groups, with a tag value assigned to each group. The GBP configuration associates subinterfaces and IP prefixes with the different groups and specifies an ACL to enforce access rules based on group tagging.

6.4.8.1 Configuration and process

This chapter describes the configuration of GBPs.

6.4.8.1.1 GBP tags in MAC-VRF/IP-VRF

In the context of a MAC-VRF, GBP tag association and enforcement only affects known unicast traffic. BUM traffic bypasses the GBP ACLs; this is irrespective of the source tag associated with the source MAC address.

In the context of an IP-VRF, GBP tag association only applies to unicast IP traffic.

The following sections describe the GBP tag association, programming, advertisement, and processing.

6.4.8.1.1.1 Local GBP tag association and programming

Local GBP tag association and programming are supported in MAC-VRFs and IP-VRFs.

GBP tagging on MAC-VRFs

GBP tags are configured at the bridged subinterface level, for example, `ethernet-1/1.1` receives the tag 10. All MAC addresses learned over that subinterface are associated with that tag in the MAC table. This is valid for dynamic or static MAC addresses.

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

YANG state is added to show the GBP tag ID in the MAC table as the 16-bit value that is programmed. This is displayed using the following command:

info from state network-instance bridge-table mac-table mac[address] group-based-policy-tag <value>

The allowed tag value range depends on the platform. The tag is programmed along with the MAC entry in the MAC table and not in the proxy-ARP/ND or ARP/ND tables.

A default tag or tag value of 0 is used in the following circumstances:

- IRB MACs do not have any GBP tag, so they have a GBP tag value of 0.
  - Routed traffic coming from a bridged subinterface receives the source tag from the Layer 2 source MAC lookup and the destination tag from a Layer 3 destination LPM lookup in the route table.
  - To advertise the IRB MACs with a specific GBP tag, an export policy matching the IRB IPs (within a prefix list) and EVPN route type 2 are added, with an action to add the GBP extended community with the needed value.
- The following internal MACs display the GBP tag N/A (or no GBP tag) in the MAC table:
  - IRB interface MACs
  - IRB interface anycast MACs
  - Layer 2 proxy anti-spoofing MACs
  - Reserved MACs
  - Eth-CFM MACs
  - IRB interface VRRP MACs
- MAC addresses learned on a subinterface that is not associated with a GBP group show a tag value of 0 in the MAC table.
- MAC addresses learned on a subinterface that is associated with a GBP group that receives a tag value of 0 also show a tag value of 0 in the MAC table.

GBP tagging on IP-VRFs

Local GBP tags (groups) are configured at the following levels:

- **routed subinterface level**

  All the local address prefixes associated with the subinterface inherit the configured GBP tag.

- **static route level**

  GBP tags are configured using the `network-instance static-routes route group-based-policy group <name>` command.

YANG state is added to show the tag ID in the route table, as the 16-bit value that is programmed. This is displayed using the following command:

info from state network-instance route-table ipv4/6-unicast route group-based-policy-tag

Default tags are used following the same principles in the MAC table. For example, local subnets configured on Layer 3 subinterfaces not associated with a GBP group have a tag value of 0 in the route table.

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

6.4.8.1.1.2 GBP tag advertisement in EVPN

The GBP tags associated with MACs, MAC/IPs, and IP Prefixes are encoded in an extended community defined in draft-lrss-bess-evpn-group-policy, and advertised along with the EVPN MAC/IP Advertisement routes and IP Prefix routes.

The following considerations apply to the GBP tag advertisement in EVPN, assuming that the `group-based-policy add-community` command is configured in the corresponding network instances:

- Advertised MAC/IP routes take the tag from the MAC table entry for the MAC address.
  - MAC/IP routes advertised for ARP/ND entries take the tag from the MAC entry, and the same applies to EVPN-IFL-HOST advertised routes.
  - On reception, the best route's tag is processed if `group-based-policy install-from-community` is configured. If a route is not the best, the GBP tag that conveys the non-selected route is ignored.
- The default tag (tag value 0) and the advertisement of routes with a tag value of 0:
  - When MAC or route entries are associated with a tag value of 0, all the BGP routes for those MAC addresses or IP prefixes will not include the GBP extended community.
  - For example, suppose you have three subinterfaces in a MAC-VRF:
    - **subif1**

      This subinterface is a static group A with a tag value of 10. All MAC addresses learned on `subif1` are advertised with the GBP extended community that includes tag value 10. This includes the MAC/IP Advertisement routes without IP, with IP, and EVPN-IFL-HOST routes advertised.
    - **subif2**

      This subinterface is static group B with a tag value of 0. All MAC addresses learned on `subif2` are advertised without the GBP extended community.
    - **subif3**

      This subinterface is static with no group. All MAC addresses learned on `subif3` are advertised without the GBP extended community.

6.4.8.1.1.3 GBP tag processing

EVPN MAC/IP Advertisement routes received with a tag value of 0 (no extended community or extended community with a value of 0) are programmed with a tag value of 0.

Nodes only support a limited size of GBP tags, typically lower than the full GBP tag size of 16 bits. Limits can also be different between Layer 2 and Layer 3 GBP tags. Therefore, received BGP routes may convey tags that exceed the size supported by the importing node:

- If the received tag falls into the supported range in the platform and network instance, then the tag is accepted and programmed.
- If the received tag does not fall into the supported range in the platform or network instance:
  - An import policy can be applied to replace the received GBP extended community with a new GBP extended community that contains a supported value. The tag is now valid and can be programmed in the MAC or route table.

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

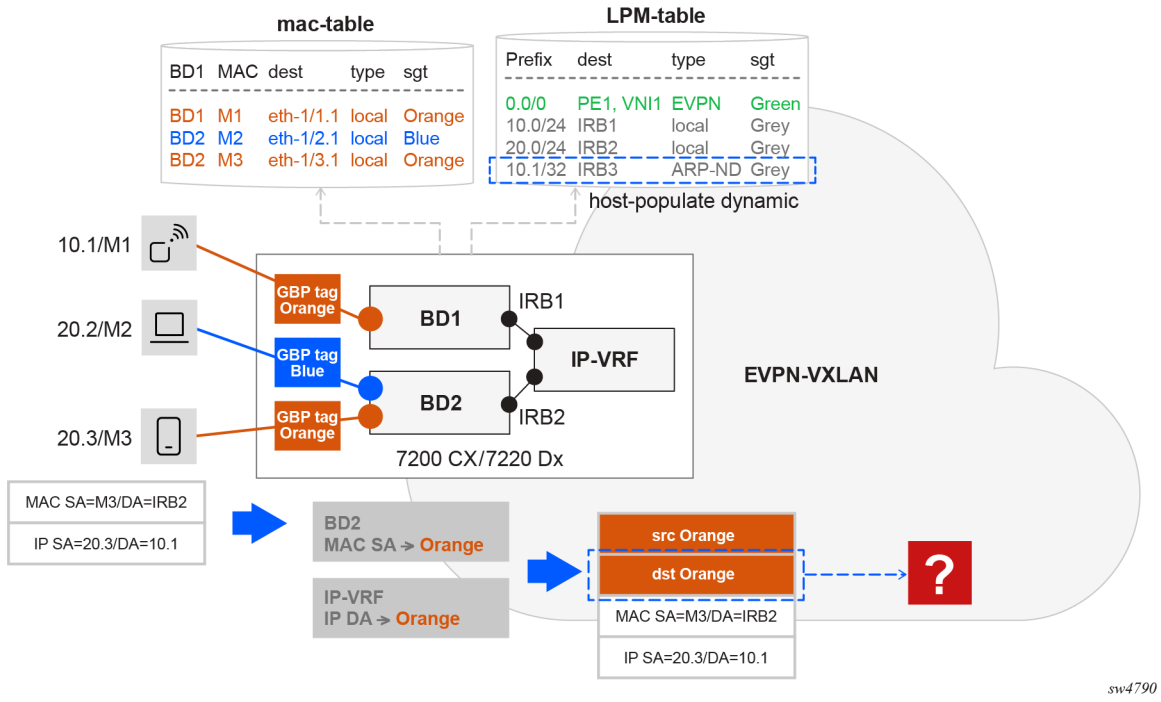
– If no import policy is applied, the MAC entry or IP route entry is programmed with a tag value of 0.

By default, GBP extended communities are propagated by ABRs and ASBRs in Model B scenarios or across multi-instance network instances, unless explicitly modified by import or export policies.

6.4.8.1.2 GBP tags in ARP/ND and host entries

Individual ARP/ND entries in the host table can be associated with a different GBP tag from the tag of their parent IRB interface. This is shown in the following figure:

Figure 57: Assignment of GBP tag to ARP/ND host routes



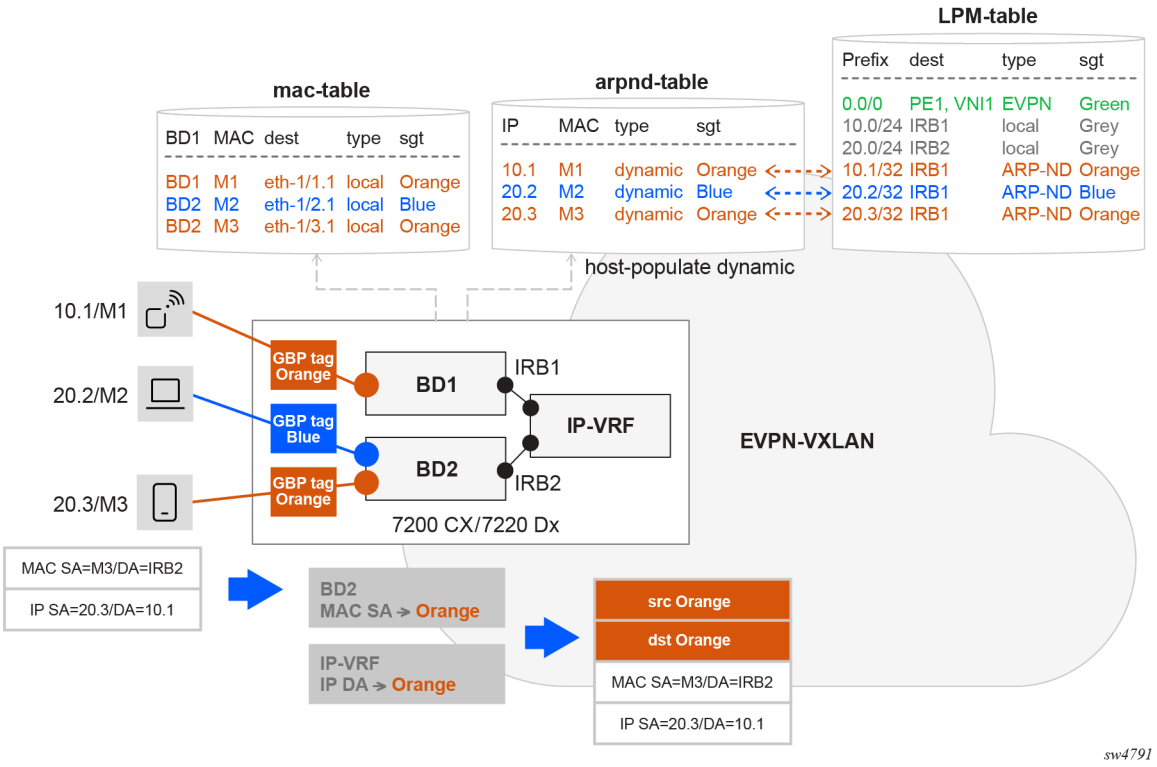
For traffic from 20.3 to 10.1, the ACL rule requires that the destination tag lookup in the IP-VRF returns 0RANGE, which corresponds to the tag assigned to 10.1's bridged subinterface.

To address this, the bridged subinterface tag (0RANGE) is propagated to the corresponding ARP/ND binds in the ARP/ND tables and ARP/ND host routes in the route table. This can only be completed when the associated ARP/ND entry MAC originates from the same bridged subinterface.

The following figure shows a higher level perspective of the process:

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

Figure 58: Propagation of tags to ARP/ND binds and host routes



By default, `arpnd_mgr` uses the GBP tag of the IRB subinterface to program the `group-based-policy-tag` of all the entries. Once a tag is associated with a MAC entry, and an ARP/ND entry is created referencing that MAC, the system links the ARP/ND binding to the existing MAC tag in the MAC table. The `arpnd_mgr` pulls that information and displays the tag in the state leaf, as follows:

```
info from state subinterface ipv4/ipv6 arp/neighbor-discovery neighbor[=ipv4/ipv6-address]
 group-based-policy-tag <gbp-tag>
```

In Figure 58: Propagation of tags to ARP/ND binds and host routes, suppose 10.1 is attached to an ORANGE bridged subinterface, however the IRB subinterface of BD1 is associated with tag GREY. When applying a GBP ACL in the IP-VRF, traffic to 10.1/32 should be associated with tag ORANGE, and not GREY. In order to program 10.1/32 in the route table with tag ORANGE, the bridged subinterface in BD1 is configured with tag ORANGE. When MAC M1 is learned on the subinterface, M1 gets tag ORANGE in the MAC table. When GBP is enabled in the IP-VRF of BD1 (and in BD1 itself), the system populates the ARP/ND entry for 10.1 with tag ORANGE.

When the `interface subinterface[=irb1] ipv4/6 arp/neighbor-discovery host-route populate dynamic` command is configured in IRB1, `arpnd_mgr` triggers the creation of an ARP/NP host route in the route table for 10.1/32, which also inherits the GBP tag ORANGE (inherited from the MAC table). When **host-route populate** is not enabled, the destination IP lookup to resolve the ARP retrieves the GBP tag from the ARP entry, so the destination tag in the example is ORANGE.

If **host-route populate** is enabled, but `interface subinterface[=irb1] ipv4/6.arp/neighbor-discovery host-route populate datapath-programming` is not, the ARP/ND host is

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

not programmed in the route table, but the destination GBP tag is obtained from a lookup on the LEM/host table to resolve the ARP.

If there is a static route for the host route in the route table and it is associated with a different GBP tag, the static route tag is used as the destination tag.

6.4.8.1.3 Bypassing the GBP ACL enforcement

The following packets bypass the GBP enforcement and are not subject to the GBP ACLs, irrespective of their MAC or IP addresses:

- **BUM traffic ingressing a bridged subinterface**

  This ensures ARP requests can be forwarded without being dropped.

- **Layer 3 traffic routed via an IRB with ARP/ND resolved but no MAC in MAC table**

  For traffic routed from the IP-VRF into the MAC-VRF via IRB, if the ARP/ND resolution exists, but the corresponding MAC entry is missing from the MAC table, the traffic bypasses the GBP ACL configured in the IP-VRF.

- **Packets routed from the MAC-VRF to the IRB**

  These packets bypass the GBP ACL applied to the MAC-VRF. Instead, packets are routed from the MAC-VRF to the IRB and are filtered by the GBP ACL configured in the IP-VRF.

- **CPM self generated traffic and extracted traffic to CPM**

  This traffic bypasses the GBP ACLs. In 7220 IXR-D*x* platforms, the extraction TCAM slice has a higher priority than the one for GBP. This ensures that all extracted traffic bypasses the GBP ACLs. The CPM traffic includes, but is not limited to:

  – ICMP

  – ARP/ND

  – routing protocols (IGP, BGP)

  – BFD

- **CPM reinjected packets**

  For example, unknown ARP Requests when the Layer 2 proxy-ARP function is enabled in the MAC-VRF. Reinjected packets are placed in the egress pipeline so that GBP ACLs cannot be applied to them.

  As a result of this behavior, for example, when a user attempts to verify reachability to a remote host using an ICMP ping, the packet may be filtered by the GBP ACL. In contrast, an ICMP trace route to the same destination may succeed because the packets are processed by all CPMs along the path.

Any other packets do not bypass the ACL enforcement when GBP is enabled in the network instance.

6.4.8.2 GBP in an EVPN distributed IRB network

GBP in an EVPN distributed IRB network enforces policies at the ingress node and allows unwanted traffic to be dropped at the point of entry. This avoids unnecessary bandwidth consumption within the network. Hosts can reside in the same subnet or across different subnets. In both cases, the system classifies the source traffic into the appropriate GBP tag and maps the destination to its corresponding GBP tag. When the source and destination GBP tags are identified, they are passed to the ACL, which uses them

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

to determine whether the traffic should be forwarded or dropped. For more information about micro-segmentation and the distributed IRB model, see the *SR Linux ACL and Traffic Steering Guide*.

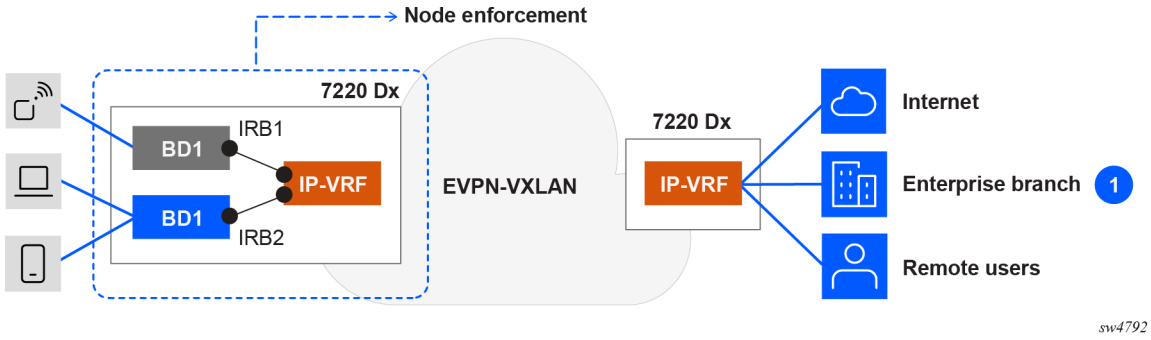
For intra-subnet communication, the source and destination GBP tags are derived from MAC address lookups in the MAC table, which stores GBP tag associations used for Layer 2 policy enforcement.

For inter-subnet communication, the source GBP tag is derived from a MAC address lookup in the MAC table. The destination GBP tag is obtained from an LPM on the corresponding IP-VRF route table.

On the 7220 IXR-D4, Layer 3 (inter-subnet) group-based policy requires the **td4-mode** parameter to be set to 2 and XDP to be restarted. For information about configuring **td4-mode**, see the "Changing datapath resource management settings" section in the *SR Linux Configuration Basics Guide*.

The following figure shows three hosts, attached to the same or different subnets and where the enforcement takes place.

Figure 59: Enforcement in an EVPN distributed IRB model



6.4.8.3 GBP in an EVPN broadcast domain or centralized IRB network

GBP in an EVPN broadcast domain or centralized IRB network is used in data centers where access nodes are connected to broadcast domains and inter-subnet communication is handled exclusively by a centralized IRB on a central node. In this architecture, the access node forwards traffic based on MAC lookups, while inter-subnet forwarding occurs at the central node. Policy enforcement must be distributed: Layer 2 enforcement is performed at the access node for intra-subnet traffic and Layer 3 enforcement is carried out at the central IRB node for inter-subnet traffic. For more information about micro-segmentation and the centralized IRB model, see the *SR Linux ACL and Traffic Steering Guide*.

For intra-subnet communication, the source and destination GBP tags are obtained from MAC address lookups in the ingress node's MAC table, which maintains GBP tag associations for Layer 2 policy enforcement.

For inter-subnet communication, the source and destination GBP tags are derived from one of the following:

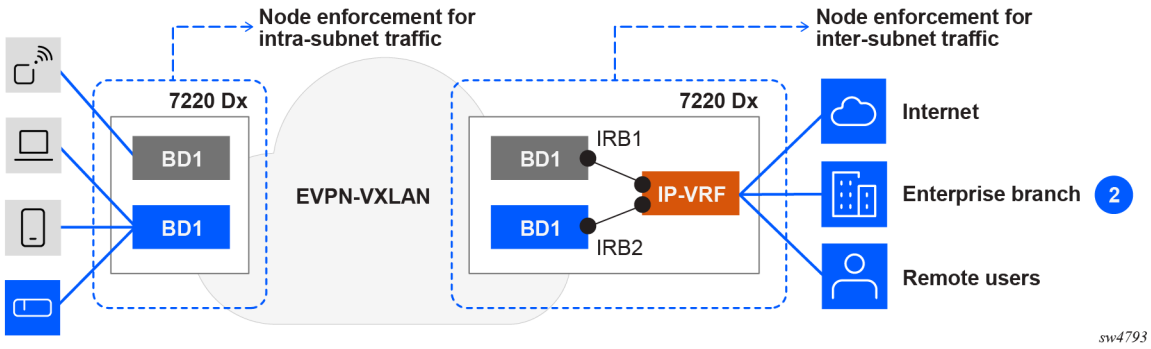
- LPM lookups of the source and destination IP addresses in the route table of the egress node
- source tag from a source MAC lookup and destination tag in the LPM table

On the 7220 IXR-D4, Layer 3 group-based policy requires the **td4-mode** parameter to be set to 2 and XDP to be restarted. For information about configuring **td4-mode**, see the "Changing datapath resource management settings" section in the *SR Linux Configuration Basics Guide*.

The following diagram illustrates this distributed enforcement.

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

Figure 60: Enforcement in an EVPN centralized IRB model



6.4.8.4 GBP configuration and state examples

This chapter contains examples of GBP configurations.

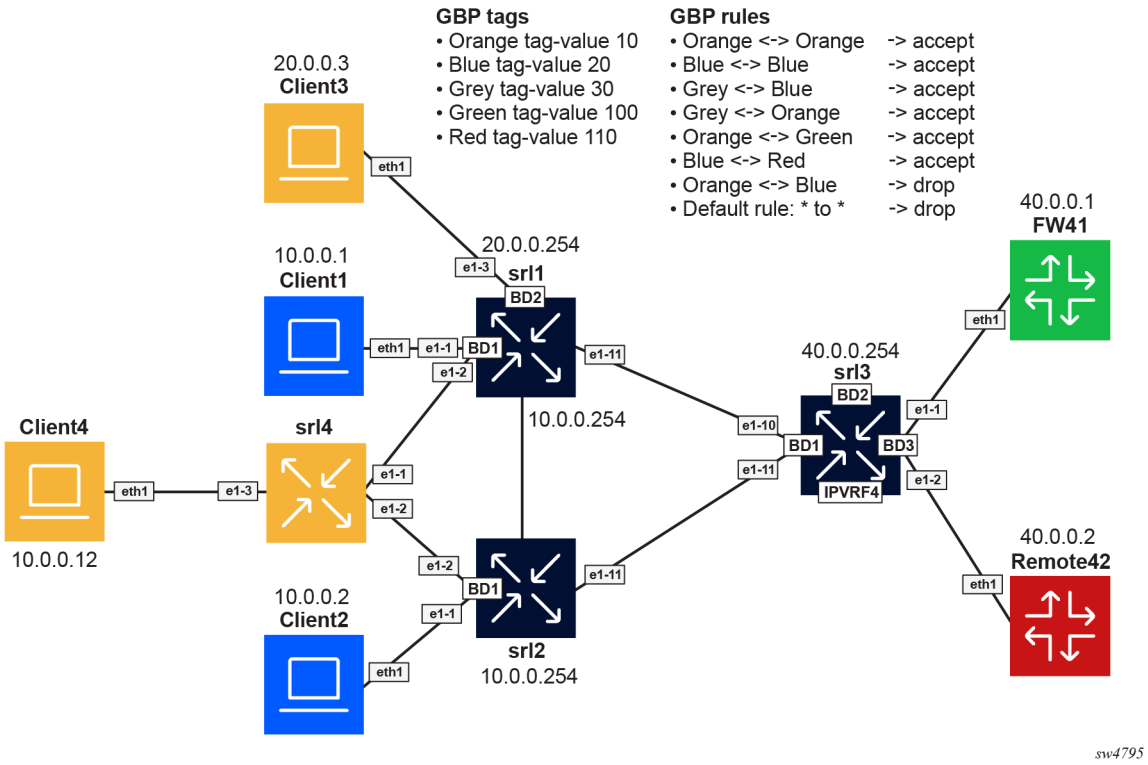
6.4.8.4.1 GBP in Layer 2 centralized IRB EVPN VXLAN example

GBP can be configured on SR Linux in a centralized IRB EVPN VXLAN scenario. For more information about micro-segmentation and this Layer 2 model, see the *SR Linux ACL and Traffic Steering Guide*. This example includes multiple SR Linux routers configured to provide connectivity while enforcing policies based on group membership.

The following topology consists of three routers (srl1, srl2, srl3) forming a mesh backbone with six clients attached.

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

Figure 61: Centralized IRB EVPN VXLAN topology



In this example, the following groups are defined, which are configured consistently across all the routers:

Table 11: Group definitions

Tag name	Tag value
ORANGE	10
BLUE	20
GREY	30
GREEN	100
RED	110

The following policies are enforced between the groups. Each row indicates a bidirectional policy between two groups. For example, the third row indicates that traffic between endpoints in the GREY and BLUE groups is accepted in both directions. The rules do not need to be bidirectional, for example, traffic from group ORANGE to group BLUE could be dropped, while traffic from group BLUE to group ORANGE may be accepted. This example uses symmetric policies for simplicity.

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

Table 12: Policy definitions

Group 1	Group 2	Action
ORANGE	ORANGE	Accept
BLUE	BLUE	Accept
GREY	BLUE	Accept
GREY	ORANGE	Accept
ORANGE	GREEN	Accept
BLUE	RED	Accept
ORANGE	BLUE	Drop

The following steps describe the configuration of GBP on the nodes.

1. Enable the following global configuration in the three routers.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat platform resource-management group-based-policy
set / platform resource-management group-based-policy mac-lookup true
set / platform resource-management group-based-policy lpm-source-lookup true
```

   The statement **mac-lookup true** enables a source MAC lookup in the MAC table to find the associated GBP tag in MAC-VRFs. This command does not require a chassis reboot.

   The statement **lpm-source-lookup true** enables the source IP lookup in the LPM table in IP-VRFs. This command requires a chassis reboot.

   The state leaf `chassis-reboot-required` under `platform resource-management group-based-policy` can be checked to verify if a reboot is needed.

2. Configure BGP EVPN to distribute and program the GBP tags in each network instance. Enable the following configuration under each BGP EVPN instance in all network instances where GBP is used.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat network-instance * protocols bgp-evpn bgp-instance 1 group-based-
policy
set / network-instance BD1 protocols bgp-evpn bgp-instance 1 group-based-policy add-
community true
set / network-instance BD1 protocols bgp-evpn bgp-instance 1 group-based-policy install-
from-community true
set / network-instance BD2 protocols bgp-evpn bgp-instance 1 group-based-policy add-
community true
set / network-instance BD2 protocols bgp-evpn bgp-instance 1 group-based-policy install-
from-community true
```

3. Define the groups on each router by configuring the groups in all network instances, even if they are not used in that network instance. This ensures consistency across the network.

```
// srl1
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat network-instance * group-based-policy group *
set / network-instance BD1 group-based-policy group BLUE tag-value 20
set / network-instance BD1 group-based-policy group GREEN tag-value 100
```


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

```
set / network-instance BD1 group-based-policy group GREY tag-value 30
set / network-instance BD1 group-based-policy group ORANGE tag-value 10
set / network-instance BD1 group-based-policy group RED tag-value 110
set / network-instance BD2 group-based-policy group BLUE tag-value 20
set / network-instance BD2 group-based-policy group GREEN tag-value 100
set / network-instance BD2 group-based-policy group GREY tag-value 30
set / network-instance BD2 group-based-policy group ORANGE tag-value 10
set / network-instance BD2 group-based-policy group RED tag-value 110

// srl2
--{ candidate shared default }--[  ]--
A:admin@srl2# info flat network-instance * group-based-policy group *
set / network-instance BD1 group-based-policy group BLUE tag-value 20
set / network-instance BD1 group-based-policy group GREEN tag-value 100
set / network-instance BD1 group-based-policy group GREY tag-value 30
set / network-instance BD1 group-based-policy group ORANGE tag-value 10
set / network-instance BD1 group-based-policy group RED tag-value 110

// srl3
--{ candidate shared default }--[  ]--
A:admin@srl3# info flat network-instance * group-based-policy group *
set / network-instance BD1 group-based-policy group BLUE tag-value 20
set / network-instance BD1 group-based-policy group GREEN tag-value 100
set / network-instance BD1 group-based-policy group GREY tag-value 30
set / network-instance BD1 group-based-policy group ORANGE tag-value 10
set / network-instance BD1 group-based-policy group RED tag-value 110
set / network-instance BD2 group-based-policy group BLUE tag-value 20
set / network-instance BD2 group-based-policy group GREEN tag-value 100
set / network-instance BD2 group-based-policy group GREY tag-value 30
set / network-instance BD2 group-based-policy group ORANGE tag-value 10
set / network-instance BD2 group-based-policy group RED tag-value 110
set / network-instance BD3 group-based-policy group BLUE tag-value 20
set / network-instance BD3 group-based-policy group GREEN tag-value 100
set / network-instance BD3 group-based-policy group GREY tag-value 30
set / network-instance BD3 group-based-policy group ORANGE tag-value 10
set / network-instance BD3 group-based-policy group RED tag-value 110
set / network-instance IPVRF4 group-based-policy group BLUE tag-value 20
set / network-instance IPVRF4 group-based-policy group GREEN tag-value 100
set / network-instance IPVRF4 group-based-policy group GREY tag-value 30
set / network-instance IPVRF4 group-based-policy group ORANGE tag-value 10
set / network-instance IPVRF4 group-based-policy group RED tag-value 110
```

4. Associate the endpoints to the groups on each router.

```
// srl1
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat network-instance * group-based-policy interface *
set / network-instance BD1 group-based-policy interface ethernet-1/1.1 group [ BLUE ]
set / network-instance BD1 group-based-policy interface ethernet-1/2.1 group [ ORANGE ]
set / network-instance BD2 group-based-policy interface ethernet-1/3.1 group [ ORANGE ]

// srl2
--{ candidate shared default }--[  ]--
A:admin@srl2# info flat network-instance * group-based-policy interface *
set / network-instance BD1 group-based-policy interface ethernet-1/1.1 group [ BLUE ]
set / network-instance BD1 group-based-policy interface ethernet-1/2.1 group [ ORANGE ]

// srl3
--{ + candidate shared default }--[  ]--
A:admin@srl3# info flat network-instance * group-based-policy interface *
set / network-instance BD3 group-based-policy interface ethernet-1/1.1 group [ GREEN ]
set / network-instance BD3 group-based-policy interface ethernet-1/2.1 group [ RED ]
set / network-instance IPVRF4 group-based-policy interface irb0.1 group [ GREY ]
set / network-instance IPVRF4 group-based-policy interface irb0.2 group [ GREY ]
```


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

```
set / network-instance IPVRF4 group-based-policy interface irb0.3 group [ GREY ]
```

5. Define the policies between the groups on each router. To simplify the example, the same policies are configured in all the routers. The same configuration is applied to srl2 and srl3 on the corresponding network instances.

```
// srl1
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat network-instance * group-based-policy acl
set / network-instance BD1 group-based-policy acl entry 10 match source-group [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 10 match destination-group
 [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 10 action accept
set / network-instance BD1 group-based-policy acl entry 20 match source-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 20 match destination-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 20 action accept
set / network-instance BD1 group-based-policy acl entry 30 match source-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 30 match destination-group [ GREY ]
set / network-instance BD1 group-based-policy acl entry 30 action accept
set / network-instance BD1 group-based-policy acl entry 40 match source-group [ GREY ]
set / network-instance BD1 group-based-policy acl entry 40 match destination-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 40 action accept
set / network-instance BD1 group-based-policy acl entry 45 match source-group [ GREY ]
set / network-instance BD1 group-based-policy acl entry 45 match destination-group
 [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 45 action accept
set / network-instance BD1 group-based-policy acl entry 47 match source-group [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 47 match destination-group [ GREY ]
set / network-instance BD1 group-based-policy acl entry 47 action accept
set / network-instance BD1 group-based-policy acl entry 50 match source-group [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 50 match destination-group [ GREEN ]
set / network-instance BD1 group-based-policy acl entry 50 action accept
set / network-instance BD1 group-based-policy acl entry 60 match source-group [ GREEN ]
set / network-instance BD1 group-based-policy acl entry 60 match destination-group
 [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 60 action accept
set / network-instance BD1 group-based-policy acl entry 70 match source-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 70 match destination-group [ RED ]
set / network-instance BD1 group-based-policy acl entry 70 action accept
set / network-instance BD1 group-based-policy acl entry 80 match source-group [ RED ]
set / network-instance BD1 group-based-policy acl entry 80 match destination-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 80 action accept
set / network-instance BD1 group-based-policy acl entry 90 match source-group [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 90 match destination-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 90 action log true
set / network-instance BD1 group-based-policy acl entry 90 action drop
set / network-instance BD1 group-based-policy acl entry 100 match source-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 100 match destination-group
 [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 100 action log true
set / network-instance BD1 group-based-policy acl entry 100 action drop
set / network-instance BD1 group-based-policy acl entry 200 action log true
set / network-instance BD1 group-based-policy acl entry 200 action drop
set / network-instance BD2 group-based-policy acl entry 10 match source-group [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 10 match destination-group
 [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 10 action accept
set / network-instance BD2 group-based-policy acl entry 20 match source-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 20 match destination-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 20 action accept
set / network-instance BD2 group-based-policy acl entry 30 match source-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 30 match destination-group [ GREY ]
set / network-instance BD2 group-based-policy acl entry 30 action accept
```


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

```
set / network-instance BD2 group-based-policy acl entry 40 match source-group [ GREY ]
set / network-instance BD2 group-based-policy acl entry 40 match destination-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 40 action accept
set / network-instance BD2 group-based-policy acl entry 45 match source-group [ GREY ]
set / network-instance BD2 group-based-policy acl entry 45 match destination-group
 [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 45 action accept
set / network-instance BD2 group-based-policy acl entry 47 match source-group [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 47 match destination-group [ GREY ]
set / network-instance BD2 group-based-policy acl entry 47 action accept
set / network-instance BD2 group-based-policy acl entry 50 match source-group [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 50 match destination-group [ GREEN ]
set / network-instance BD2 group-based-policy acl entry 50 action accept
set / network-instance BD2 group-based-policy acl entry 60 match source-group [ GREEN ]
set / network-instance BD2 group-based-policy acl entry 60 match destination-group
 [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 60 action accept
set / network-instance BD2 group-based-policy acl entry 70 match source-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 70 match destination-group [ RED ]
set / network-instance BD2 group-based-policy acl entry 70 action accept
set / network-instance BD2 group-based-policy acl entry 80 match source-group [ RED ]
set / network-instance BD2 group-based-policy acl entry 80 match destination-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 80 action accept
set / network-instance BD2 group-based-policy acl entry 90 match source-group [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 90 match destination-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 90 action log true
set / network-instance BD2 group-based-policy acl entry 90 action drop
set / network-instance BD2 group-based-policy acl entry 100 match source-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 100 match destination-group
 [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 100 action log true
set / network-instance BD2 group-based-policy acl entry 100 action drop
set / network-instance BD2 group-based-policy acl entry 200 action log true
set / network-instance BD2 group-based-policy acl entry 200 action drop
```

Show and state commands

The following show commands display the GBP tags associated with each MAC.

```
--{ + running }--[  ]--
A:admin@srl1# show network-instance BD1 bridge-table mac-table all
-----------------------------------------------------------------------------------------------
Mac-table of network instance BD1
-----------------------------------------------------------------------------------------------
+---------+---------------+--------+--------+------+-----+----------+--------+--------------+
|Address  |    Destination |   Dest |  Type  |Active|Aging|   Not-   |  GBP   |  Last Update |
|         |                |  Index |        |      |     |Programmed|  Tags  |              |
|         |                |        |        |      |     |  Reason  |        |              |
+=========+===============+========+========+======+=====+==========+========+==============+
|00:CA:CA:| reserved       | 0      |reserved| false| N/A | reserved | N/A    |2025-11-26T20:|
|DE:BA:CA |                |        |        |      |     |          |        | 23:46.000Z   |
|         |                |        |        |      |     |          |        |              |
|1A:2D:08:|vxlan-interface:|86914466| evpn-  | true | N/A | none     | 30     |2025-11-26T20:|
|FF:00:41 |vxlan0.1        |        | static |      |     |          |        |24:05.000Z    |
|         |vtep:100.0.0.3  | 1308   |        |      |     |          |        |              |
|         |vni:1           |        |        |      |     |          |        |              |
|         |                |        |        |      |     |          |        |              |
|1A:BE:06:| reserved       | 0      |reserved| false| N/A | reserved | N/A    |2025-11-26T20:|
|FF:00:00 |                |        |        |      |     |          |        |23:46.000Z    |
|         |                |        |        |      |     |          |        |              |
|1A:E6:07:|vxlan-interface:|86914466| evpn-  | true | N/A | none     | 0      |2025-11-26T20:|
|FF:00:00 |vxlan0.1        |        | static |      |     |          |        |23:57.000Z    |
|         |vtep:100.0.0.2  |1304    |        |      |     |          |        |              |
```


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

```
|         |vni:1           |        |        |      |     |          |        |              |
|         |                |        |        |      |     |          |        |              |
|AA:C1:AB:| ethernet-1/2.1 | 2      | evpn   | true | N/A | none     | 10     |2025-11-26T20:|
|A7:F1:20 |                |        |        |      |     |          |(ORANGE)| 39:39.000Z   |
|         |                |        |        |      |     |          |        |              |
|AA:C1:AB:| ethernet-1/1.1 | 1      | learnt | true | 300 | none     | 20     |2025-11-26T20:|
|DF:05:9E |                |        |        |      |     |          | (BLUE) | 40:12.000Z   |
+---------+----------------+--------+--------+------+-----+----------+--------+------+
Total Irb Macs                 :    0 Total    0 Active
Total Static Macs              :    0 Total    0 Active
Total Duplicate Macs           :    0 Total    0 Active
Total Learnt Macs              :    1 Total    1 Active
Total Evpn Macs                :    1 Total    1 Active
Total Evpn static Macs         :    2 Total    2 Active
Total Irb anycast Macs         :    0 Total    0 Active
Total Proxy Antispoof Macs     :    0 Total    0 Active
Total Reserved Macs            :    2 Total    0 Active
Total Eth-cfm Macs             :    0 Total    0 Active
Total Irb Vrrps                :    0 Total    0 Active

--{ candidate shared default }--[  ]--
A:admin@srl1# show network-instance BD1 bridge-table mac-table mac 1A:70:07:FF:00:41
-----------------------------------------------------------------------------------------
Mac-table of network instance BD1
-----------------------------------------------------------------------------------------
Mac                     : 1A:70:07:FF:00:41
Destination             : vxlan-interface:vxlan0.1 vtep:100.0.0.2 vni:1
Dest Index              : 867613807248
Type                    : evpn-static
Programming Status      : Success
Aging                   : N/A
Last Update             : 2025-11-26T16:08:48.000Z
Duplicate Detect time   : N/A
Hold down time remaining: N/A
GBP Tags                : 0
-----------------------------------------------------------------------------------------

--{ candidate shared default }--[  ]--
A:admin@srl1# show network-instance BD1 bridge-table mac-table mac AA:C1:AB:D8:B8:6E
-----------------------------------------------------------------------------------------
Mac-table of network instance BD1
-----------------------------------------------------------------------------------------
Mac                     : AA:C1:AB:D8:B8:6E
Destination             : ethernet-1/2.1
Dest Index              : 2
Type                    : learnt
Programming Status      : Success
Aging                   : 292
Last Update             : 2025-11-26T19:53:51.000Z
Duplicate Detect time   : N/A
Hold down time remaining: N/A
GBP Tags                : 10 (ORANGE)
-----------------------------------------
```

The following state commands display the GBP tags associated with the ARP/ND entries in the host table.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info from state interface irb0 subinterface 1 ipv4 arp neighbor * | as table
+--------+-------+----------+-----------------+---------+-----------------+---------+----------+
| Inter- | Sub-  | Ipv4-    |    Link-layer-  | Origin  | Expiration-time | Group-  | Datapath-|
| face   | inter-| address  |    address      |         |                 | based-  | programm-|
|        | face  |          |                 |         |                 | policy- | ing      |
|        |       |          |                 |         |                 | tag     | status   |
```


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

```
|        |       |          |                 |         |                 |        |          |
|        |       |          |                 |         |                 |        |          |

|        |       |          |                 |         |                 |        |          |
|        |       |          |                 |         |                 |        |          |
|        |       |          |                 |         |                 |        |          |
+========+=======+==========+=================+=========+=================+========+==========+
| irb0   |     1 | 10.0.0.1 |AA:C1:AB:E3:B4:CF| dynamic | 2025-11-        |     20 | success  |
|        |       |          |                 |         | 26T23:53:48.808Z|        |          |
|        |       |          |                 |         | (3 hours from   |        |          |
|        |       |          |                 |         | now)            |        |          |
| irb0   |     1 | 10.0.0.2 |AA:C1:AB:0F:24:1E| evpn    |                 |     20 | success  |
| irb0   |     1 | 10.0.0.12|AA:C1:AB:D8:B8:6E| dynamic | 2025-11-        |     10 | success  |
|        |       |          |                 |         | 26T23:53:56.753Z|        |          |
|        |       |          |                 |         | (3 hours from   |        |          |
|        |       |          |                 |         | now)            |        |          |
+--------+-------+----------+-----------------+---------+-----------------+--------+----------+

--{ candidate shared default }--[  ]--
A:admin@srl1# info from state interface irb0 subinterface 1 ipv4 arp neighbor * group-based-
policy-tag | as table
+---------------------+-------------+-----------------+------------------------+
|      Interface      | Subinterface |   Ipv4-address   | Group-based-policy-tag |
+=====================+=============+=================+========================+
| irb0                |           1 | 10.0.0.1        |                     20 |
| irb0                |           1 | 10.0.0.2        |                     20 |
| irb0                |           1 | 10.0.0.12       |                     10 |
+---------------------+-------------+-----------------+------------------------+
```

The following state commands display the GBP tags associated with the routes in the IP routing table.

Note: The **show network instance <name> ipv4/ipv6 route gbp-tag .*** command displays similar information.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info from state network-instance IPVRF4 route-table ipv4-unicast route * id *
 route-type * route-owner * origin-network-instance * group-based-policy-tag | as table
+--------+--------------+----+------------------+----------------------+----------+--------+
|Network-| Ipv4-prefix  | Id |    Route-type    |   Route-owner        | Origin-  | Group- |
|instance|              |    |                  |                      | network- | based- |
|        |              |    |                  |                      | instance | policy-|
|        |              |    |                  |                      |          | tag    |
|        |              |    |                  |                      |          |        |
+========+==============+====+==================+======================+==========+========+
| IPVRF4 | 10.0.0.0/24  |  0 | bgp-evpn         | bgp_evpn_mgr         | IPVRF4   |     30 |
| IPVRF4 | 10.0.0.0/24  |  6 | local            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 10.0.0.2/32  |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr | IPVRF4   |     20 |
| IPVRF4 | 10.0.0.254/32 |  6 | host            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 10.0.0.255/32 |  6 | host            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 20.0.0.0/24  |  7 | local            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 20.0.0.254/32 |  7 | host            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 20.0.0.255/32 |  7 | host            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 40.0.0.0/24  |  0 | bgp-evpn         | bgp_evpn_mgr         | IPVRF4   |     30 |
+--------+--------------+----+------------------+----------------------+----------+--------+

--{ candidate shared default }--[  ]--
A:admin@srl3# info from state network-instance IPVRF4 route-table ipv4-unicast route * id *
 route-type * route-owner * origin-network-instance * group-based-policy-tag | as table
+--------+--------------+----+------------------+----------------------+----------+--------+
|Network-|  Ipv4-prefix | Id |    Route-type    |  Route-owner         | Origin-  | Group- |
|instance|              |    |                  |                      | network- | based- |
|        |              |    |                  |                      | instance | policy-|
|        |              |    |                  |                      |          | tag    |
```


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

```
|        |               |    |                   |                        |          |        |
+========+===============+====+===================+========================+==========+========+
| IPVRF4 | 10.0.0.0/24   |  0 | bgp-evpn          | bgp_evpn_mgr           | IPVRF4   |     30 |
| IPVRF4 | 10.0.0.1/32   |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr  | IPVRF4   |     20 |
| IPVRF4 | 10.0.0.2/32   |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr  | IPVRF4   |     20 |
| IPVRF4 | 10.0.0.12/32  |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr  | IPVRF4   |     10 |
| IPVRF4 | 20.0.0.0/24   |  0 | bgp-evpn          | bgp_evpn_mgr           | IPVRF4   |     30 |
| IPVRF4 | 20.0.0.3/32   |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr  | IPVRF4   |     10 |
| IPVRF4 | 40.0.0.0/24   |  5 | local             | net_inst_mgr           | IPVRF4   |     30 |
| IPVRF4 | 40.0.0.254/32 |  5 | host              | net_inst_mgr           | IPVRF4   |     30 |
| IPVRF4 | 40.0.0.255/32 |  5 | host              | net_inst_mgr           | IPVRF4   |     30 |
+--------+---------------+----+-------------------+------------------------+----------+--------+
```

The following show command displays the GBP ACL statistics.

```
--{ candidate shared default }--[  ]--
A:admin@srl3# show acl gbp-filter IPVRF4 entry 200
=====================================================================
Network Instance: IPVRF4
Entries         : 13
---------------------------------------------------------------------
Entry 200
  Match               : protocol=<undefined>, [any](*)->[any](*)
  Action              : drop
  Collect Stats       : true
  Match Packets       : 135
  Last Match          : 3 seconds ago
---------------------------------------------------------------------
```

6.4.8.4.2 GBP in Layer 3 distributed IRB EVPN VXLAN example

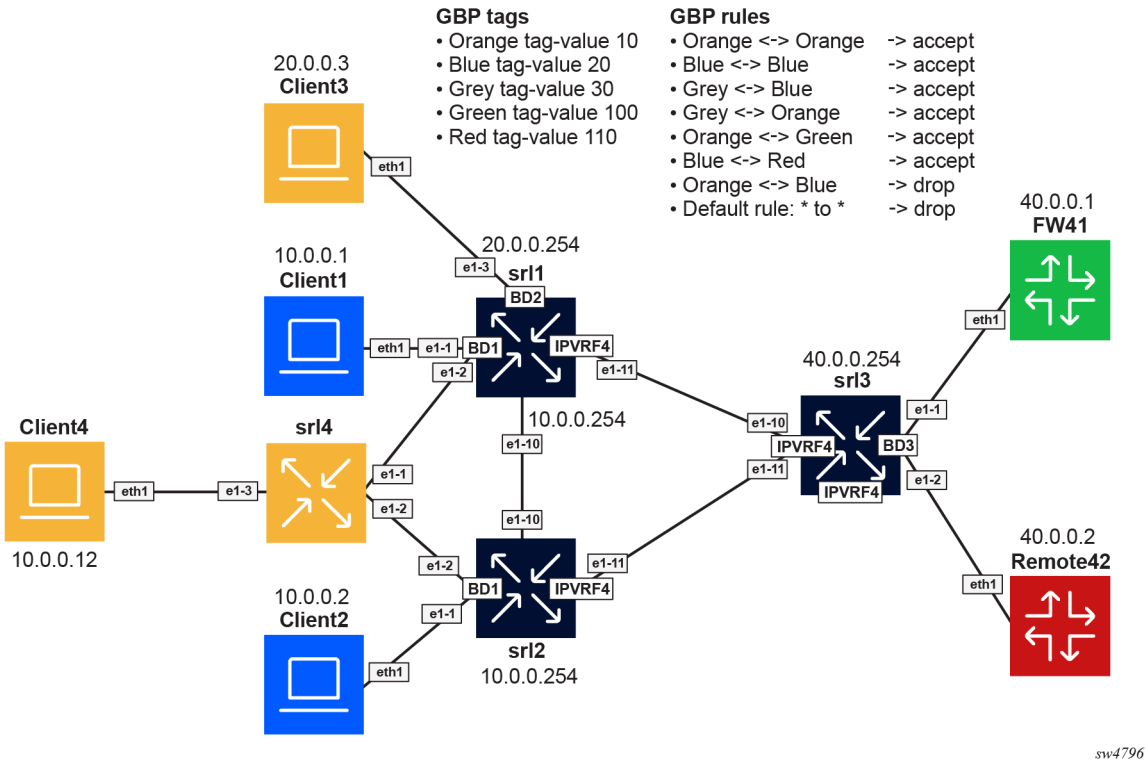
GBP can be configured on SR Linux in a Layer 3 distributed IRB EVPN VXLAN model. For more information about micro-segmentation and this Layer 3 model, see the *SR Linux ACL and Traffic Steering Guide*. This example includes multiple SR Linux routers configured to provide Layer 3 connectivity between different broadcast domains while enforcing policies based on group membership.

On the 7220 IXR-D4, Layer 3 group-based policy requires the **td4-mode** parameter to be set to 2 and XDP to be restarted. For information about configuring **td4-mode**, see the "Changing datapath resource management settings" section in the *SR Linux Configuration Basics Guide*.

The following topology consists of three routers (srl1, srl2, srl3) and one access switch (srl4) forming a mesh backbone with six clients attached.

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

Figure 62: Distributed IRB EVPN VXLAN topology



In this example, the following groups are defined, which are configured consistently across all the routers:

Table 13: Group definitions

Tag name	Tag value
ORANGE	10
BLUE	20
GREY	30
GREEN	100
RED	110

The following policies are enforced between the groups. Each row indicates a bidirectional policy between two groups. For example, the third row indicates that traffic between endpoints in the GREY and BLUE groups is accepted in both directions. The rules do not need to be bidirectional, for example, traffic from group ORANGE to group BLUE could be dropped, while traffic from group BLUE to group ORANGE may be accepted. This example uses symmetric policies for simplicity.

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

Table 14: Policy definitions

Group 1	Group 2	Action
ORANGE	ORANGE	Accept
BLUE	BLUE	Accept
GREY	BLUE	Accept
GREY	ORANGE	Accept
ORANGE	GREEN	Accept
BLUE	RED	Accept
ORANGE	BLUE	Drop

The following steps describe the configuration of GBP on the nodes.

1. Enable the following global configuration in the three routers.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat platform resource-management group-based-policy
set / platform resource-management group-based-policy mac-lookup true
set / platform resource-management group-based-policy lpm-source-lookup true
```

   The statement **mac-lookup true** enables a source MAC lookup in the MAC table to find the associated GBP tag in MAC-VRFs. This command does not require a chassis reboot.

   The statement **lpm-source-lookup true** enables the source IP lookup in the LPM table in IP-VRFs. This command requires a chassis reboot.

   The state leaf `chassis-reboot-required` under `platform resource-management group-based-policy` can be checked to verify if a reboot is needed.

2. Configure BGP EVPN to distribute and program the GBP tags in each network instance. Enable the following configuration under each BGP EVPN instance in all network instances where GBP is used.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat network-instance * protocols bgp-evpn bgp-instance 1 group-based-
policy
set / network-instance BD1 protocols bgp-evpn bgp-instance 1 group-based-policy add-
community true
set / network-instance BD1 protocols bgp-evpn bgp-instance 1 group-based-policy install-
from-community true
set / network-instance BD2 protocols bgp-evpn bgp-instance 1 group-based-policy add-
community true
set / network-instance BD2 protocols bgp-evpn bgp-instance 1 group-based-policy install-
from-community true
set / network-instance IPVRF4 protocols bgp-evpn bgp-instance 1 group-based-policy add-
community true
set / network-instance IPVRF4 protocols bgp-evpn bgp-instance 1 group-based-policy install-
from-community true
```

3. Define the groups on each router by configuring the groups in all network instances, even if they are not used in that network instance. This ensures consistency across the network.

```
// srl1
```


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

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat network-instance * group-based-policy group *
set / network-instance BD1 group-based-policy group BLUE tag-value 20
set / network-instance BD1 group-based-policy group GREEN tag-value 100
set / network-instance BD1 group-based-policy group GREY tag-value 30
set / network-instance BD1 group-based-policy group ORANGE tag-value 10
set / network-instance BD1 group-based-policy group RED tag-value 110
set / network-instance BD2 group-based-policy group BLUE tag-value 20
set / network-instance BD2 group-based-policy group GREEN tag-value 100
set / network-instance BD2 group-based-policy group GREY tag-value 30
set / network-instance BD2 group-based-policy group ORANGE tag-value 10
set / network-instance BD2 group-based-policy group RED tag-value 110
set / network-instance IPVRF4 group-based-policy group BLUE tag-value 20
set / network-instance IPVRF4 group-based-policy group GREEN tag-value 100
set / network-instance IPVRF4 group-based-policy group GREY tag-value 30
set / network-instance IPVRF4 group-based-policy group ORANGE tag-value 10
set / network-instance IPVRF4 group-based-policy group RED tag-value 110

// srl2
--{ candidate shared default }--[  ]--
A:admin@srl2# info flat network-instance * group-based-policy group *
set / network-instance BD1 group-based-policy group BLUE tag-value 20
set / network-instance BD1 group-based-policy group GREEN tag-value 100
set / network-instance BD1 group-based-policy group GREY tag-value 30
set / network-instance BD1 group-based-policy group ORANGE tag-value 10
set / network-instance BD1 group-based-policy group RED tag-value 110
set / network-instance IPVRF4 group-based-policy group BLUE tag-value 20
set / network-instance IPVRF4 group-based-policy group GREEN tag-value 100
set / network-instance IPVRF4 group-based-policy group GREY tag-value 30
set / network-instance IPVRF4 group-based-policy group ORANGE tag-value 10
set / network-instance IPVRF4 group-based-policy group RED tag-value 110

// srl3
--{ candidate shared default }--[  ]--
A:admin@srl3# info flat network-instance * group-based-policy group *
set / network-instance BD3 group-based-policy group BLUE tag-value 20
set / network-instance BD3 group-based-policy group GREEN tag-value 100
set / network-instance BD3 group-based-policy group GREY tag-value 30
set / network-instance BD3 group-based-policy group ORANGE tag-value 10
set / network-instance BD3 group-based-policy group RED tag-value 110
set / network-instance IPVRF4 group-based-policy group BLUE tag-value 20
set / network-instance IPVRF4 group-based-policy group GREEN tag-value 100
set / network-instance IPVRF4 group-based-policy group GREY tag-value 30
set / network-instance IPVRF4 group-based-policy group ORANGE tag-value 10
set / network-instance IPVRF4 group-based-policy group RED tag-value 110
```

4. Associate the endpoints to the groups on each router.

```
// srl1
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat network-instance * group-based-policy interface *
set / network-instance BD1 group-based-policy interface ethernet-1/1.1 group [ BLUE ]
set / network-instance BD1 group-based-policy interface ethernet-1/2.1 group [ ORANGE ]
set / network-instance BD2 group-based-policy interface ethernet-1/3.1 group [ ORANGE ]
set / network-instance IPVRF4 group-based-policy interface irb0.1 group [ GREY ]
set / network-instance IPVRF4 group-based-policy interface irb0.2 group [ GREY ]

// srl2
--{ candidate shared default }--[  ]--
A:admin@srl2# info flat network-instance * group-based-policy interface *
set / network-instance BD1 group-based-policy interface ethernet-1/1.1 group [ BLUE ]
set / network-instance BD1 group-based-policy interface ethernet-1/2.1 group [ ORANGE ]
set / network-instance IPVRF4 group-based-policy interface irb0.1 group [ GREY ]
```


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

```
// srl3
--{ candidate shared default }--[  ]--
A:admin@srl3# info flat network-instance * group-based-policy interface *
set / network-instance BD3 group-based-policy interface ethernet-1/1.1 group [ GREEN ]
set / network-instance BD3 group-based-policy interface ethernet-1/2.1 group [ RED ]
set / network-instance IPVRF4 group-based-policy interface irb0.3 group [ GREY ]
```

5. Define the policies between the groups on each router. To simplify the example, the same policies are configured in all the routers. The same configuration is applied to srl2 and srl3 on the corresponding network instances.

```
// srl1
--{ candidate shared default }--[  ]--
A:admin@srl1# info flat network-instance * group-based-policy acl
set / network-instance BD1 group-based-policy acl entry 10 match source-group [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 10 match destination-group
 [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 10 action accept
set / network-instance BD1 group-based-policy acl entry 20 match source-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 20 match destination-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 20 action accept
set / network-instance BD1 group-based-policy acl entry 30 match source-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 30 match destination-group [ GREY ]
set / network-instance BD1 group-based-policy acl entry 30 action accept
set / network-instance BD1 group-based-policy acl entry 40 match source-group [ GREY ]
set / network-instance BD1 group-based-policy acl entry 40 match destination-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 40 action accept
set / network-instance BD1 group-based-policy acl entry 45 match source-group [ GREY ]
set / network-instance BD1 group-based-policy acl entry 45 match destination-group
 [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 45 action accept
set / network-instance BD1 group-based-policy acl entry 47 match source-group [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 47 match destination-group [ GREY ]
set / network-instance BD1 group-based-policy acl entry 47 action accept
set / network-instance BD1 group-based-policy acl entry 50 match source-group [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 50 match destination-group [ GREEN ]
set / network-instance BD1 group-based-policy acl entry 50 action accept
set / network-instance BD1 group-based-policy acl entry 60 match source-group [ GREEN ]
set / network-instance BD1 group-based-policy acl entry 60 match destination-group
 [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 60 action accept
set / network-instance BD1 group-based-policy acl entry 70 match source-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 70 match destination-group [ RED ]
set / network-instance BD1 group-based-policy acl entry 70 action accept
set / network-instance BD1 group-based-policy acl entry 80 match source-group [ RED ]
set / network-instance BD1 group-based-policy acl entry 80 match destination-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 80 action accept
set / network-instance BD1 group-based-policy acl entry 90 match source-group [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 90 match destination-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 90 action log true
set / network-instance BD1 group-based-policy acl entry 90 action drop
set / network-instance BD1 group-based-policy acl entry 100 match source-group [ BLUE ]
set / network-instance BD1 group-based-policy acl entry 100 match destination-group
 [ ORANGE ]
set / network-instance BD1 group-based-policy acl entry 100 action log true
set / network-instance BD1 group-based-policy acl entry 100 action drop
set / network-instance BD1 group-based-policy acl entry 200 action log true
set / network-instance BD1 group-based-policy acl entry 200 action drop
set / network-instance BD2 group-based-policy acl entry 10 match source-group [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 10 match destination-group
 [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 10 action accept
set / network-instance BD2 group-based-policy acl entry 20 match source-group [ BLUE ]
```


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

```
set / network-instance BD2 group-based-policy acl entry 20 match destination-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 20 action accept
set / network-instance BD2 group-based-policy acl entry 30 match source-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 30 match destination-group [ GREY ]
set / network-instance BD2 group-based-policy acl entry 30 action accept
set / network-instance BD2 group-based-policy acl entry 40 match source-group [ GREY ]
set / network-instance BD2 group-based-policy acl entry 40 match destination-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 40 action accept
set / network-instance BD2 group-based-policy acl entry 45 match source-group [ GREY ]
set / network-instance BD2 group-based-policy acl entry 45 match destination-group
 [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 45 action accept
set / network-instance BD2 group-based-policy acl entry 47 match source-group [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 47 match destination-group [ GREY ]
set / network-instance BD2 group-based-policy acl entry 47 action accept
set / network-instance BD2 group-based-policy acl entry 50 match source-group [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 50 match destination-group [ GREEN ]
set / network-instance BD2 group-based-policy acl entry 50 action accept
set / network-instance BD2 group-based-policy acl entry 60 match source-group [ GREEN ]
set / network-instance BD2 group-based-policy acl entry 60 match destination-group
 [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 60 action accept
set / network-instance BD2 group-based-policy acl entry 70 match source-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 70 match destination-group [ RED ]
set / network-instance BD2 group-based-policy acl entry 70 action accept
set / network-instance BD2 group-based-policy acl entry 80 match source-group [ RED ]
set / network-instance BD2 group-based-policy acl entry 80 match destination-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 80 action accept
set / network-instance BD2 group-based-policy acl entry 90 match source-group [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 90 match destination-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 90 action log true
set / network-instance BD2 group-based-policy acl entry 90 action drop
set / network-instance BD2 group-based-policy acl entry 100 match source-group [ BLUE ]
set / network-instance BD2 group-based-policy acl entry 100 match destination-group
 [ ORANGE ]
set / network-instance BD2 group-based-policy acl entry 100 action log true
set / network-instance BD2 group-based-policy acl entry 100 action drop
set / network-instance BD2 group-based-policy acl entry 200 action log true
set / network-instance BD2 group-based-policy acl entry 200 action drop
set / network-instance IPVRF4 group-based-policy acl entry 10 match source-group [ ORANGE ]
set / network-instance IPVRF4 group-based-policy acl entry 10 match destination-group
 [ ORANGE ]
set / network-instance IPVRF4 group-based-policy acl entry 10 action accept
set / network-instance IPVRF4 group-based-policy acl entry 20 match source-group [ BLUE ]
set / network-instance IPVRF4 group-based-policy acl entry 20 match destination-group
 [ BLUE ]
set / network-instance IPVRF4 group-based-policy acl entry 20 action accept
set / network-instance IPVRF4 group-based-policy acl entry 30 match source-group [ BLUE ]
set / network-instance IPVRF4 group-based-policy acl entry 30 match destination-group
 [ GREY ]
set / network-instance IPVRF4 group-based-policy acl entry 30 action accept
set / network-instance IPVRF4 group-based-policy acl entry 40 match source-group [ GREY ]
set / network-instance IPVRF4 group-based-policy acl entry 40 match destination-group
 [ BLUE ]
set / network-instance IPVRF4 group-based-policy acl entry 40 action accept
set / network-instance IPVRF4 group-based-policy acl entry 45 match source-group [ GREY ]
set / network-instance IPVRF4 group-based-policy acl entry 45 match destination-group
 [ ORANGE ]
set / network-instance IPVRF4 group-based-policy acl entry 45 action accept
set / network-instance IPVRF4 group-based-policy acl entry 47 match source-group [ ORANGE ]
set / network-instance IPVRF4 group-based-policy acl entry 47 match destination-group
 [ GREY ]
set / network-instance IPVRF4 group-based-policy acl entry 47 action accept
set / network-instance IPVRF4 group-based-policy acl entry 50 match source-group [ ORANGE ]
```


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

```
set / network-instance IPVRF4 group-based-policy acl entry 50 match destination-group
 [ GREEN ]
set / network-instance IPVRF4 group-based-policy acl entry 50 action accept
set / network-instance IPVRF4 group-based-policy acl entry 60 match source-group [ GREEN ]
set / network-instance IPVRF4 group-based-policy acl entry 60 match destination-group
 [ ORANGE ]
set / network-instance IPVRF4 group-based-policy acl entry 60 action accept
set / network-instance IPVRF4 group-based-policy acl entry 70 match source-group [ BLUE ]
set / network-instance IPVRF4 group-based-policy acl entry 70 match destination-group
 [ RED ]
set / network-instance IPVRF4 group-based-policy acl entry 70 action accept
set / network-instance IPVRF4 group-based-policy acl entry 80 match source-group [ RED ]
set / network-instance IPVRF4 group-based-policy acl entry 80 match destination-group
 [ BLUE ]
set / network-instance IPVRF4 group-based-policy acl entry 80 action accept
set / network-instance IPVRF4 group-based-policy acl entry 90 match source-group [ ORANGE ]
set / network-instance IPVRF4 group-based-policy acl entry 90 match destination-group
 [ BLUE ]
set / network-instance IPVRF4 group-based-policy acl entry 90 action log true
set / network-instance IPVRF4 group-based-policy acl entry 90 action drop
set / network-instance IPVRF4 group-based-policy acl entry 100 match source-group [ BLUE ]
set / network-instance IPVRF4 group-based-policy acl entry 100 match destination-group
 [ ORANGE ]
set / network-instance IPVRF4 group-based-policy acl entry 100 action log true
set / network-instance IPVRF4 group-based-policy acl entry 100 action drop
set / network-instance IPVRF4 group-based-policy acl entry 200 action log true
set / network-instance IPVRF4 group-based-policy acl entry 200 action drop
```

Show and state commands

The following show commands display the GBP tags associated with each MAC.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# show network-instance BD1 bridge-table mac-table all
-----------------------------------------------------------------------------------------------
Mac-table of network instance BD1
-----------------------------------------------------------------------------------------------
+-------------------+------------------+--------------+------------+--------+-------+---------+
|      Address      |    Destination   |  Dest Index  |    Type    | Active | Aging |         |
|                   |                  |              |            |        |       | GBP Tags|
|                   |                  |              |            |        |       |         |
+===================+==================+==============+============+========+=======+=========+
| 00:00:5E:00:01:01 | irb-interface    | 0            | irb-       | true   | N/A   | N/A     |
|                   |                  |              | interface- |        |       |         |
|                   |                  |              | anycast    |        |       |         |
| 1A:70:07:FF:00:41 | vxlan-interface: | 867613807248 | evpn-static| true   | N/A   | 0       |
|                   | vxlan0.1         |              |            |        |       |         |
|                   | vtep:100.0.0.2   |              |            |        |       |         |
|                   | vni:1            |              |            |        |       |         |
| 1A:B3:06:FF:00:41 | irb-interface    | 0            | irb        | true   | N/A   | N/A     |
|                   |                  |              | interface  |        |       |         |
|                   |                  |              |            |        |       |         |
| AA:C1:AB:0F:24:1E | vxlan-interface: | 867613807248 | evpn       | true   | N/A   | 20      |
|                   | vxlan0.1         |              |            |        |       |         |
|                   | vtep:100.0.0.2   |              |            |        |       |         |
|                   | vni:1            |              |            |        |       |         |
| AA:C1:AB:D8:B8:6E | ethernet-1/2.1   | 2            | learnt     | true   | 300   | 10      |
|                   |                  |              |            |        |       |(ORANGE) |
|                   |                  |              |            |        |       |         |
| AA:C1:AB:E3:B4:CF | ethernet-1/1.1   | 1            | learnt     | true   | 300   | 20      |
|                   |                  |              |            |        |       |(BLUE)   |
+-------------------+------------------+--------------+------------+--------+-------+---------+
Total Irb Macs                :    1 Total    1 Active
```


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

```
Total Static Macs              :    0 Total    0 Active
Total Duplicate Macs           :    0 Total    0 Active
Total Learnt Macs              :    2 Total    2 Active
Total Evpn Macs                :    1 Total    1 Active
Total Evpn static Macs         :    1 Total    1 Active
Total Irb anycast Macs         :    1 Total    1 Active
Total Proxy Antispoof Macs     :    0 Total    0 Active
Total Reserved Macs            :    0 Total    0 Active
Total Eth-cfm Macs             :    0 Total    0 Active
Total Irb Vrrps                :    0 Total    0 Active

--{ candidate shared default }--[  ]--
A:admin@srl1# show network-instance BD1 bridge-table mac-table mac 1A:70:07:FF:00:41
-------------------------------------------------------------------------------------------------
Mac-table of network instance BD1
-------------------------------------------------------------------------------------------------
Mac                     : 1A:70:07:FF:00:41
Destination             : vxlan-interface:vxlan0.1 vtep:100.0.0.2 vni:1
Dest Index              : 867613807248
Type                    : evpn-static
Programming Status      : Success
Aging                   : N/A
Last Update             : 2025-11-26T16:08:48.000Z
Duplicate Detect time   : N/A
Hold down time remaining: N/A
GBP Tags                : 0
-------------------------------------------------------------------------------------------------

--{ candidate shared default }--[  ]--
A:admin@srl1# show network-instance BD1 bridge-table mac-table mac AA:C1:AB:D8:B8:6E
-------------------------------------------------------------------------------------------------
Mac-table of network instance BD1
-------------------------------------------------------------------------------------------------
Mac                     : AA:C1:AB:D8:B8:6E
Destination             : ethernet-1/2.1
Dest Index              : 2
Type                    : learnt
Programming Status      : Success
Aging                   : 292
Last Update             : 2025-11-26T19:53:51.000Z
Duplicate Detect time   : N/A
Hold down time remaining: N/A
GBP Tags                : 10 (ORANGE)
-----------------------------------------
```

The following state commands display the GBP tags associated with the ARP/ND entries in the host table.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info from state interface irb0 subinterface 1 ipv4 arp neighbor * | as table
+--------+-------+----------+-----------------+---------+-----------------+--------+----------+
| Inter- | Sub-  | Ipv4-    |    Link-layer-  | Origin  | Expiration-time | Group- | Datapath-|
| face   | inter-| address  |    address      |         |                 | based- | programm-|
|        | face  |          |                 |         |                 | policy-| ing      |
|        |       |          |                 |         |                 | tag    | status   |
|        |       |          |                 |         |                 |        |          |
|        |       |          |                 |         |                 |        |          |

|        |       |          |                 |         |                 |        |          |
|        |       |          |                 |         |                 |        |          |
|        |       |          |                 |         |                 |        |          |
+========+=======+==========+=================+=========+=================+========+==========+
| irb0   |     1 | 10.0.0.1 |AA:C1:AB:E3:B4:CF| dynamic | 2025-11-        |     20 | success  |
|        |       |          |                 |         | 26T23:53:48.808Z|        |          |
```


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

```
|        |       |          |                 |         | (3 hours from   |        |          |
|        |       |          |                 |         | now)            |        |          |
| irb0   |     1 | 10.0.0.2 |AA:C1:AB:0F:24:1E| evpn    |                 |     20 | success  |
| irb0   |     1 | 10.0.0.12|AA:C1:AB:D8:B8:6E| dynamic | 2025-11-        |     10 | success  |
|        |       |          |                 |         | 26T23:53:56.753Z|        |          |
|        |       |          |                 |         | (3 hours from   |        |          |
|        |       |          |                 |         | now)            |        |          |
+--------+-------+----------+-----------------+---------+-----------------+--------+----------+

--{ candidate shared default }--[  ]--
A:admin@srl1# info from state interface irb0 subinterface 1 ipv4 arp neighbor * group-based-
policy-tag | as table
+---------------------+-------------+-----------------+------------------------+
|      Interface      | Subinterface |   Ipv4-address   | Group-based-policy-tag |
+=====================+=============+=================+========================+
| irb0                |           1 | 10.0.0.1        |                     20 |
| irb0                |           1 | 10.0.0.2        |                     20 |
| irb0                |           1 | 10.0.0.12       |                     10 |
+---------------------+-------------+-----------------+------------------------+
```

The following state commands display the GBP tags associated with the routes in the IP routing table.

Note: The **show network instance <name> ipv4/ipv6 route gbp-tag .*** command displays similar information.

```
--{ candidate shared default }--[  ]--
A:admin@srl1# info from state network-instance IPVRF4 route-table ipv4-unicast route * id *
 route-type * route-owner * origin-network-instance * group-based-policy-tag | as table
+--------+--------------+----+------------------+----------------------+----------+--------+
|Network-| Ipv4-prefix  | Id |    Route-type    |   Route-owner        | Origin-  | Group- |
|instance|              |    |                  |                      | network- | based- |
|        |              |    |                  |                      | instance | policy-|
|        |              |    |                  |                      |          | tag    |
|        |              |    |                  |                      |          |        |
+========+==============+====+==================+======================+==========+========+
| IPVRF4 | 10.0.0.0/24  |  0 | bgp-evpn         | bgp_evpn_mgr         | IPVRF4   |     30 |
| IPVRF4 | 10.0.0.0/24  |  6 | local            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 10.0.0.2/32  |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr | IPVRF4   |     20 |
| IPVRF4 | 10.0.0.254/32 |  6 | host            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 10.0.0.255/32 |  6 | host            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 20.0.0.0/24  |  7 | local            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 20.0.0.254/32 |  7 | host            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 20.0.0.255/32 |  7 | host            | net_inst_mgr         | IPVRF4   |     30 |
| IPVRF4 | 40.0.0.0/24  |  0 | bgp-evpn         | bgp_evpn_mgr         | IPVRF4   |     30 |
+--------+--------------+----+------------------+----------------------+----------+--------+

--{ candidate shared default }--[  ]--
A:admin@srl3# info from state network-instance IPVRF4 route-table ipv4-unicast route * id *
 route-type * route-owner * origin-network-instance * group-based-policy-tag | as table
+--------+--------------+----+------------------+----------------------+----------+--------+
|Network-|  Ipv4-prefix | Id |    Route-type    |  Route-owner         | Origin-  | Group- |
|instance|              |    |                  |                      | network- | based- |
|        |              |    |                  |                      | instance | policy-|
|        |              |    |                  |                      |          | tag    |
|        |              |    |                  |                      |          |        |
+========+==============+====+==================+======================+==========+========+
| IPVRF4 | 10.0.0.0/24  |  0 | bgp-evpn         | bgp_evpn_mgr         | IPVRF4   |     30 |
| IPVRF4 | 10.0.0.1/32  |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr | IPVRF4   |     20 |
| IPVRF4 | 10.0.0.2/32  |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr | IPVRF4   |     20 |
| IPVRF4 | 10.0.0.12/32 |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr | IPVRF4   |     10 |
| IPVRF4 | 20.0.0.0/24  |  0 | bgp-evpn         | bgp_evpn_mgr         | IPVRF4   |     30 |
| IPVRF4 | 20.0.0.3/32  |  0 | bgp-evpn-ifl-host | bgp_evpn_ifl_host_mgr | IPVRF4   |     10 |
| IPVRF4 | 40.0.0.0/24  |  5 | local            | net_inst_mgr         | IPVRF4   |     30 |
```


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

```
| IPVRF4 | 40.0.0.254/32 |  5 | host              | net_inst_mgr          | IPVRF4   |     30 |
| IPVRF4 | 40.0.0.255/32 |  5 | host              | net_inst_mgr          | IPVRF4   |     30 |
+--------+---------------+----+-------------------+-----------------------+----------------------+
```

The following show command displays the GBP ACL statistics.

```
--{ candidate shared default }--[  ]--
A:admin@srl3# show acl gbp-filter IPVRF4 entry 200
=====================================================================
Network Instance: IPVRF4
Entries         : 13
---------------------------------------------------------------------
Entry 200
  Match               : protocol=<undefined>, [any](*)->[any](*)
  Action              : drop
  Collect Stats       : true
  Match Packets       : 135
  Last Match          : 3 seconds ago
---------------------------------------------------------------------
```

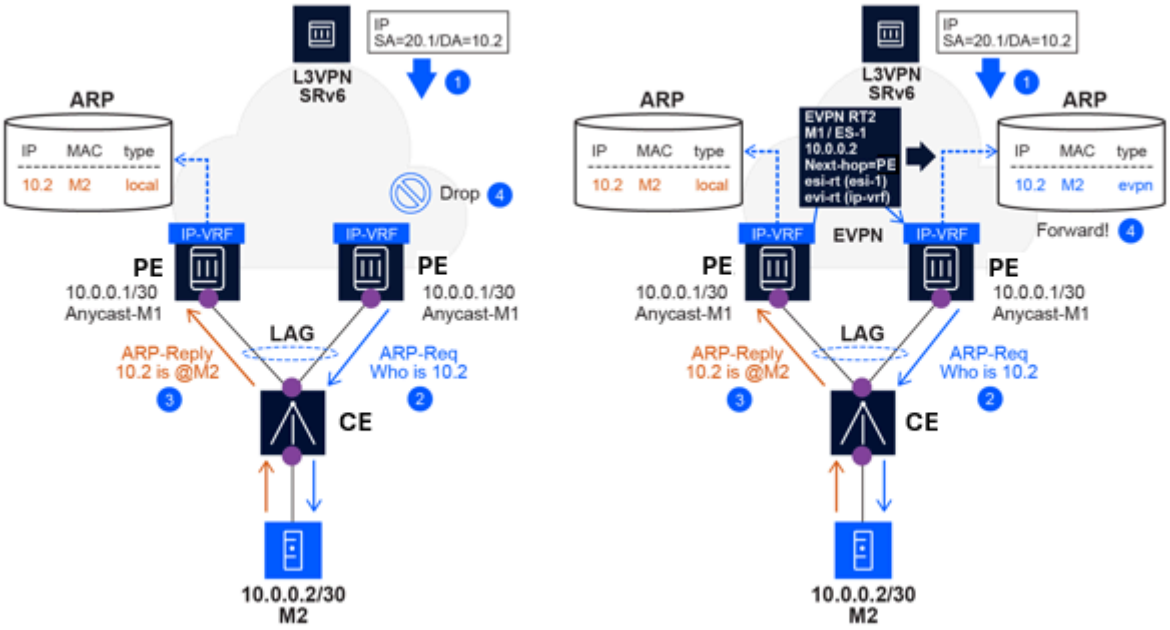
6.5 EVPN ARP/ND synchronization on Layer 3 interfaces

Note: This feature is supported on 7250 IXR (excluding mixed) platforms.

In typical access networks, the use of a single LAG to multi-home a CE to two Layer 3 PE routers is a common configuration. However, this approach presents challenges related to the synchronization of ARP/ND entries across the multihomed PEs, as illustrated in the following figure.

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

Figure 63: Multi-homed PEs with and without EVPN ARP/ND synchronization enabled



In the left-hand diagram, no synchronization of ARP/ND entries occurs between the PEs, which can lead to dropped traffic as follows:

- Only the left-hand PE learns the ARP entry for 10.0.0.2 because of how the CE hashes traffic.
- If downstream traffic arrives at the right-hand PE, it does not have a resolved entry for 10.0.0.2 and issues an ARP request.
- If the ARP request is again hashed to the left-hand PE, the right-hand PE never receives a reply and is unable to resolve the 10.0.0.2 destination IP address.
- As a result, the right-hand PE silently drops the downstream traffic.

In the right-hand diagram, this problem is addressed using synchronization of ARP/ND entries with EVPN between the multi-homed PEs (based on approach 2 in draft-ietf-bess-evpn-l3mh-proto). The solution operates as follows:

- A BGP EVPN session is established between the PEs.
- ARP/ND entries are synchronized between the two PEs using EVPN MAC/IP Advertisement routes carrying Ethernet Segment Identifier (ESI), ES-import route target, and EVI route target information.
- If downstream traffic arrives at the right-hand PE, it now has the ARP entry for 10.0.0.2 and can forward the traffic without issue.
- IP-VRF Layer 3 interfaces are used to connect to the CE LAG ports, with no requirement for MAC-VRFs or IRBs on the PEs.

This solution applies to VXLAN, MPLS BGP EVPN and IP-VPN services, and the default network instance. It supports up to four PEs on the same Ethernet Segment and interoperates with other platforms that implement the same standard.

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

Configuration prerequisites

As a prerequisite, multihomed PEs must have BGP EVPN enabled at the BGP level, either through a direct BGP neighbor relationship or via a route reflector (RR). This requirement applies whether or not EVPN is used in any service. This is necessary because EVPN is used to synchronize ARP/ND entries between the multihomed PEs.

Configuration

ARP/ND synchronization on Layer 3 interfaces requires configuration in two places:

1. The routing instance where synchronization is needed
2. The Ethernet Segments connected to the CEs for which synchronization is required

Example: Configure Layer 3 EVPN ARP/ND synchronization and associated parameters in the network instance

To configure the synchronization of ARP and ND entries on all non-IRB Layer 3 interfaces on a network instance (default or IP-VRF), use the **network-instance protocols layer-3-evpn-neighbor-synchronization** command.

```
--{ candidate shared default }--[  ]--
# info with-context network-instance default protocols layer-3-evpn-neighbor-
synchronization
    network-instance default {
        protocols {
            layer-3-evpn-neighbor-synchronization {
                admin-state enable
                route-distinguisher 10.4.4.4:6000
                evi-route-target evi-target:17676:6000
                retention-timer 300
                internal-tags {
                    set-tag-set [
                        set-1
                    ]
                }
            }
        }
    }
```

This command includes the following parameters:

- **route-distinguisher** (mandatory)

  Contrary to the BGP-VPN route distinguisher, the neighbor synchronization **route-distinguisher** command does not support autoderivation and must always be configured. In addition:

  – The RDs must be unique per routing instance.

  – The RDs referenced in **layer-3-evpn-neighbor-synchronization** must not be used in any other EVPN instance.

- **evi-route-target** (mandatory)

  Unlike regular BGP-VPN route targets, the **evi-route-target** value must be unique across network instances.

- **retention-timer**

  The **retention-timer** determines how long a synchronized ARP/ND entry of type EVPN is retained after the EVPN route is withdrawn by the remote ES peer, as follows:

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

  - When the EVPN MAC/IP Advertisement route that originated an ARP-ND entry is withdrawn, the retention timer is started.
  - The system attempts to refresh the entry by sending an ARP request or a neighbor solicitation (NS) message to the owner of the IP until the retention timer expires.
  - When the timer expires, the entry is removed.

- **internal-tag**

  Similar to IRB subinterfaces, the **internal-tag** command allows MAC/IP routes generated for the purpose of ARP/ND synchronization to be matched on export policies.

Example: Enable Layer 3 ARP/ND synchronization on the EVPN Ethernet Segment

To enable Layer 3 ARP/ND synchronization on the Ethernet Segment associated with the Layer 3 interfaces, use the **layer-3-neighbor-synchronization true** command. This command makes Layer 3 services in the ES subject to synchronization.

```
--{ candidate shared default }--[  ]--
# info with-context system network-instance protocols evpn ethernet-segments bgp-instance
 1 ethernet-segment ESI-value layer-3-neighbor-synchronization
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment ESI-value {
                                layer-3-neighbor-synchronization true
                            }
                        }
                    }
                }
            }
        }
    }
```

The following Ethernet Segment configuration considerations apply:

- **ethernet-segment type** must be **none**
- Only non-virtual Ethernet Segment type is supported
- The ES can only be associated with an interface and not with a next-hop
- Multi-homing mode can be set to any mode, but it is expected as all-active for Layer 3 interfaces. Even if Layer 3 interfaces are configured as single-active, they are not impacted by the DF election procedures.
- Routed subinterfaces and Ethernet interfaces are oper-up regardless of the Ethernet Segment multi-homing mode.
- ESI label allocation depends on the value of **use-esi-label** (**true** or **false**).
- The DF election commands have no impact on the state of Layer 3 interfaces.
- The **routes next-hop** and **routes ethernet-segment originating-ip** commands under **ethernet-segments** are supported.
- The **ethernet-segments timers activation-timer** command does not impact these Ethernet Segments for Layer 3 interfaces (since there is no DF election for Layer 3 services), but the

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

ethernet-segments timers boot-timer works the same as for any ES: the ES route is advertised after the boot timer expires.

- Up to four nodes can be attached to the same ES used for Layer 3 neighbor synchronization.
- The **ethernet-segment admin-state disable** does not bring down the Layer 3 interface.

MAC/IP Advertisement routes for ARP/ND synchronization

When EVPN ARP/ND synchronization is enabled at the network instance level, it triggers the advertisement of the MAC/IP Advertisement routes for the static or dynamic ARP/ND entries on the Layer 3 subinterface.

MAC/IP routes are advertised with the following attributes:

- the RD configured under **layer-3-evpn-neighbor-synchronization**
- the ESI of the associated Ethernet Segment
- the Ethernet tag ID of the Layer 3 interface encapsulation (which can only be 0 or VLAN-ID X)
- the MAC/IP binding of the ARP/ND static/dynamic entry
- Label-1 = 0 and no Label-2
- Extended communities:
  - ES-import extended community derived from the ESI, which ensures that MAC/IP routes are not imported in PEs that are not attached to the same ES.
  - EVI-RT extended community derived from the configured **layer-3-evpn-neighbor-synchronization evi-route-target**. This is the same extended community used in EVPN route types 7 and 8. The EVI-RT extended community value, along with the ESI, is used to identify the network instance context and interface for which the ARP/ND information must be synchronized.
- No encapsulation extended community is advertised with these routes.
- No ARP/ND extended community is advertised with these routes.

Ethernet Segment route advertisement with Layer 3 synchronization

When Layer 3 ARP/ND synchronization is enabled on the Ethernet Segment:

- The ES route includes the configured ESI, next hop and originating IP of the Ethernet Segment, as well as the ES-import extended community.
- The ES route is advertised when the associated interface transitions to oper-up, and it is withdrawn when the interface transitions to oper-down. (This is not the case in regular Ethernet Segments, where the ES route is advertised with the first subinterface up. It is the AD per ES route that is advertised based on the interface oper-state.)

Programming of ARP/ND entries

ARP/ND synchronization enables the advertisement of EVPN ARP/ND synchronization routes (type 2) for all static and dynamic ARP/ND entries. No ARP/ND entry types other than static and dynamic are synchronized. The synchronized ARP or ND entries on Layer 3 interfaces display with **origin evpn**.

Programming of ARP/ND entries on the Layer 3 interface is based on the imported MAC/IP routes or locally received ARP/ND messages.

- The MAC/IP routes used for ARP/ND synchronization are imported based on the **evi-route-target** matching the one configured in the network instance.

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

- The ESI and Ethernet Tag ID encoded in the route determine the Layer 3 subinterface in which the ARP/ND entry is programmed.
- The ESI determines the interface and the Ethernet Tag ID specifies the VLAN ID associated with the Layer 3 subinterface.
- This means multiple tagged Layer 3 interfaces per ESI and router context are supported.

As with any ARP/ND entry of type **evpn**, these entries are subject to all the procedures supported for the EVPN entries. For instance, the command **interface ethernet-1/1 subinterface 0 ipv4 arp host-route populate evpn** creates ARP/ND host routes for these entries.

For IPv6 and ND synchronization, Link-Local Addresses are synchronized as well.

Ethernet Tag ID in EVPN MAC/IP routes

The support for multiple Layer 3 interfaces per router context and per ES is based on the use of the Ethernet Tag ID to encode the VLAN ID associated with the Layer 3 interface.

The Ethernet Tag ID encoded in the MAC/IP route used for synchronization is autoderived from the Layer 3 interface encapsulation 802.1Q tags and does not involve CLI configuration. For correct operation, the Layer 3 interface encapsulation 802.1Q tags must be configured consistently in all the PEs attached to the same ES.

When a new ARP/ND entry is learned locally, it is linked to only those Layer 3 interfaces (or LAGs) that have the following encapsulation types:

- **vlan-tagging false**
- **single-tagged vlan-id X**

No other Layer 3 interface encapsulations are supported.

The use of the Ethernet Tag ID in the EVPN MAC/IP routes used for synchronization allows the synchronization of multiple IP-MAC bindings with the same value in different Layer 3 subinterfaces, as in the following example:

- Some router CEs connected to different Layer 3 subinterfaces within the same network instance use the same local Link-Local Address (LLA) and the same or different MAC addresses.
- This leads to a situation where the same Layer 3 network instance context contains multiple ND entries with the same LLA (using the same or different MACs), although each interface has only one such entry.
- This behavior is supported. Each entry is advertised in a separate MAC/IP Advertisement route, with each route tied to a different ESI/Ethernet Tag ID.
- The case where the same VLAN is used in different Ethernet Segments does not support the same IP and MAC. For example:
  - IP1/MAC1 is learned on interface ethernet-1/1 with VLAN-ID 100 and ESI-1 and on interface ethernet-1/2 with VLAN-ID 100 and ESI-2, in the same IP-VRF.
  - In this case, BGP only advertises a single MAC/IP route for IP1/MAC1, because the route key of the two routes is the same.

Route selection

- A local IP/MAC dynamic entry is always preferred over an EVPN entry, irrespective of the MAC address of the entry.

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

- If multiple MAC/IP routes with the same IP (but different route keys) are received for the same ES, only one is selected for IP/MAC programming. The selection is based on the following, in priority order:
  1. lowest RD route
  2. lowest MAC address

Support of the same IP and MAC addresses on PE interfaces

In all PEs attached to the same ES, the same IP and MAC addresses are required on the Layer 3 interfaces of the same Ethernet Segment. On 7250 IXR platforms, you can configure the same MAC address on all ES PEs via configuration of VRRP or using the **interface ethernet mac-address** command, as shown in the following example output.

```
--{ candidate shared default }--[  ]--
# info with-context interface ethernet-1/11
    interface ethernet-1/11 {
        admin-state enable
        vlan-tagging true
        ethernet {
            mac-address 00:00:5E:00:53:EF
        }
    }
# Enable Layer 3 ARP/ND synchronization on the EVPN Ethernet Segment
```


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

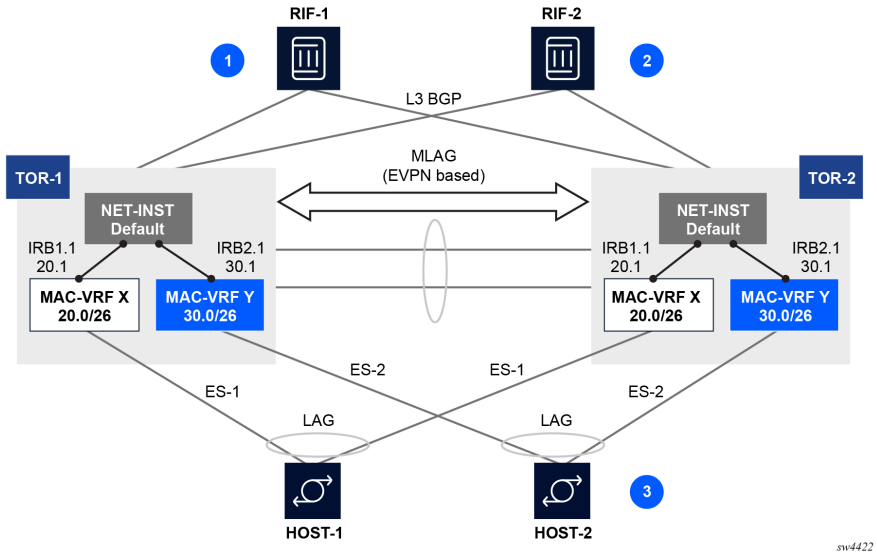
7 EVPN configuration examples

This chapter contains examples of configurations that use EVPN features.

7.1 All-active redundant connectivity example

The following figure shows an example of using EVPN multihoming as a standalone and self-contained multichassis solution.

Figure 64: Active-active connectivity example



This example uses the following features:

1. **redundancy**

   TOR redundancy is based on an all-active redundancy model.

2. **Layer 3 connectivity**

   - An anycast gateway solution is used on the IRB subinterfaces so that upstream traffic is always routed locally on the TOR receiving the traffic.
   - South-to-north traffic is sent to the active link and routed by the local IRB subinterface. In case of failure on TOR-1, TOR-2 is ready to forward on the anycast gateway IRB, without the need to wait for any VRRP protocol to converge.
   - North-to-south traffic is load-balanced by the fabric to the two TORs. ARP/ND entries are synchronized across both TORs. Host routes could be optionally created and advertised in BGP from the directly connected TOR to avoid tromboning in the downstream direction.

3. **Layer 2 connectivity**

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

- Servers do not need to run any xSTP protocols. The NDF TOR brings down the port and signals LOS to the server.
- This solution places no requirements on the servers.

This example imposes the use of a LAG on the server. The LAG can use LACP or not. The servers are unaware that the LAG is connected to two systems instead of only one.

7.1.1 Configuration for all-active connectivity example

Leaf 1 in Figure 64: Active-active connectivity example has the following configuration. Leaf 2 would have an equivalent configuration.

```
--{ [FACTORY] +* candidate shared default }--[  ]--
A:Leaf-1# info with-context
// two IRB interfaces with anycast-gw configuration is added
interface irb1 {
    subinterface 1 {
        ipv4 {
            admin-state enable
            address 20.0.0.1/24 {
                anycast-gw true
                primary
            }
            arp {
                learn-unsolicited true
                evpn {
                    advertise dynamic {
                    }
                }
            }
        }
        anycast-gw {
        }
    }
}
interface irb2 {
    subinterface 1 {
        ipv4 {
            admin-state enable
            address 30.0.0.1/24 {
                anycast-gw true
                primary
            }
            arp {
                learn-unsolicited true
                evpn {
                    advertise dynamic {
                    }
                }
            }
        }
        anycast-gw {
        }
    }
}
// lags associated with the ethernet-segments.
 In this case static, but they can be lacp based too.
interface lag1 {
    admin-state enable
    vlan-tagging true
```


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

```
    subinterface 1 {
        vlan {
            encap {
                single-tagged {
                    vlan-id 20
                }
            }
        }
    }
    lag {
        lag-type static
    }
}
interface lag2 {
    admin-state enable
    vlan-tagging true
    subinterface 1 {
        vlan {
            encap {
                single-tagged {
                    vlan-id 30
                }
            }
        }
    }
    lag {
        lag-type static
    }
}
// ES configuration
system {
    network-instance {
        protocols {
            evpn {
                ethernet-segments {
                    bgp-instance 1 {
                        ethernet-segment ES-1 {
                            admin-state enable
                            esi 00:01:00:00:00:00:00:00:00:01
                            interface lag1
                        }
                        ethernet-segment ES-2 {
                            admin-state enable
                            esi 00:02:00:00:00:00:00:00:00:02
                            interface lag2
                        }
                    }
                }
            }
        }
    }
}
// MAC-VRFs
network-instance MAC-VRF-X {
    type mac-vrf
    admin-state enable
    interface irb1.1 {
    }
    interface lag1.1 {
    }
    vxlan-interface vxlan1.20 {
    }
    protocols {
        bgp-evpn {
```


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

```
            bgp-instance 1 {
                admin-state enable
                vxlan-interface vxlan1.20
                evi 20
            }
        }
        bgp-vpn {
            bgp-instance 1 {
            }
        }
    }
}
network-instance MAC-VRF-Y {
    type mac-vrf
    admin-state enable
    interface irb2.1 {
    }
    interface lag2.1 {
    }
    vxlan-interface vxlan1.30 {
    }
    protocols {
        bgp-evpn {
            bgp-instance 1 {
                admin-state enable
                vxlan-interface vxlan1.30
                evi 30
            }
        }
        bgp-vpn {
            bgp-instance 1 {
            }
        }
    }
}
// default network-instance configuration
network-instance default {
    type default
    admin-state enable
    description "Default network instance"
    router-id 1.1.1.1
    interface ethernet-1/1.1 {
    }
    interface ethernet-1/20.1 {
    }
    interface irb1.1 {
    }
    interface irb2.1 {
    }
    interface system0.0 {
    }
    protocols {
        bgp {
            admin-state enable
            afi-safi ipv4-unicast {
                admin-state enable
            }
            autonomous-system 1234
            router-id 1.1.1.1
            group eBGP-spines {
                admin-state enable
                export-policy export-all
                peer-as 4567
                afi-safi ipv4-unicast {
```


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

```
                    admin-state enable
                }
            }
            group evpn-mh {
                admin-state enable
                export-policy export-all
                peer-as 1234
                afi-safi evpn {
                    admin-state enable
                }
            }
            neighbor 2.2.2.2 {
                admin-state enable
                peer-group evpn-mh
            }
            neighbor 10.1.4.4 {
                admin-state enable
                peer-group eBGP-spines
            }

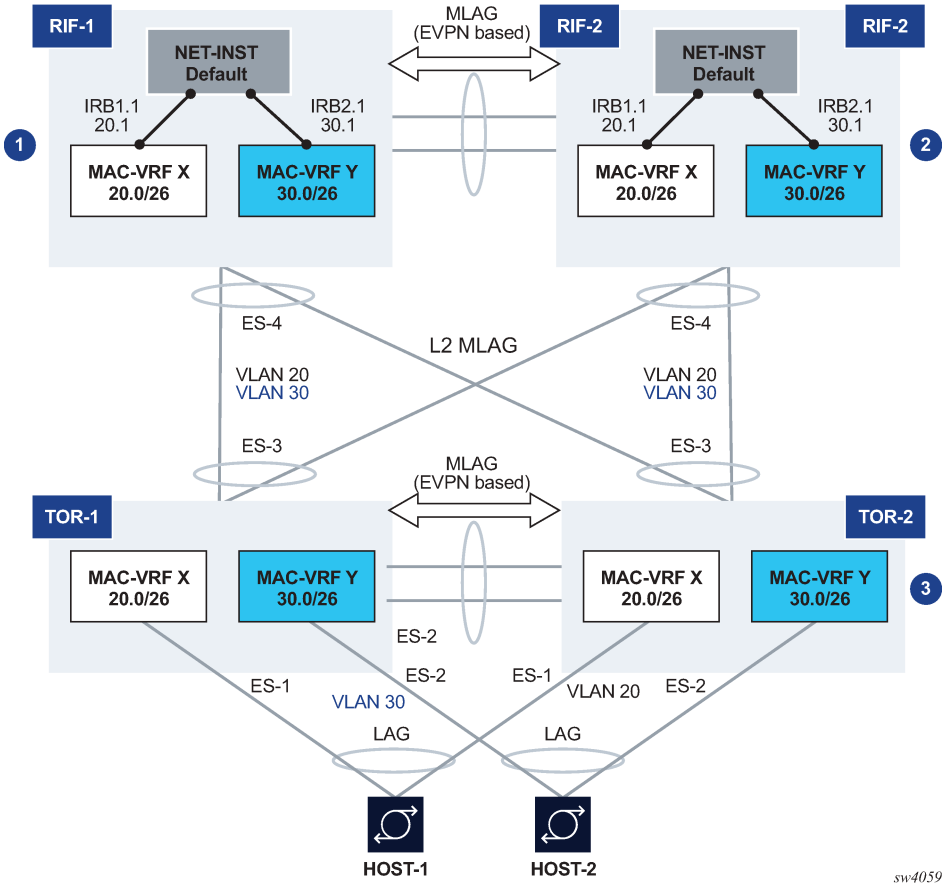
        }
    }
}
// vxlan interfaces for the MH configuration
tunnel-interface vxlan1 {
    vxlan-interface 20 {
        type bridged
        ingress {
            vni 20
        }
    }
    vxlan-interface 30 {
        type bridged
        ingress {
            vni 30
        }
    }
}
```

7.2 Hierarchical active-active connectivity example

The following figure shows an example that makes use of a Layer 2 and a Layer 3 multihoming solution.

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

Figure 65: Hierarchical active-active connectivity example



This example uses the following features:

- **Leaf and Spine configuration**
  - There are two multichassis pairs at the leaf and the spine levels.
  - The leaf pair runs all-active multihoming on its own ESs.

    The access hosts or spines are separate and independent ESs. The figure shows three ESs, one per host at the access, and one for the connectivity to the spine layer.

    Note: This is only one ES in spite of the number of spine nodes, which could be 2 or 4.

  - The spine layer runs all-active multihoming with a single ES to the leaf layer, associated with the LAG attached to the leaf layer.
  - The two tiers are independent; there are no control plane protocols between them, except for LACP if used.

    This means that the leaf and spine layer could theoretically be of a different vendor.
- **Layer 3 connectivity**

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

  - An anycast gateway solution is used on the spine layer, on the IRB subinterfaces, so that upstream traffic is always routed locally on the leaf receiving the traffic.
  - South-to-north flows are sent to only one link at a time, so there is not duplication.
  - North-to-south traffic is load-balanced by the fabric to the two spines and for the spines load-balanced to the leafs. ARP/ND entries are synchronized across both spines. Host routes can optionally be created and advertised in BGP from the directly connected spine to avoid tromboning in the downstream direction.
- **Layer 2 connectivity**
  - Hosts are running LAG with or without LACP.
  - Leafs and spines are connected by standard Layer 2 LAGs that carry the VLANs for the two broadcast domains shown in Figure 65: Hierarchical active-active connectivity example.

7.2.1 Configuration for hierarchical active-active connectivity example

Spine 1 in Figure 65: Hierarchical active-active connectivity example has the following configuration. The configuration for Spine 2 would be equivalent.

```
--{ [FACTORY] +* candidate shared default }--[  ]--
A:Spine-1# info with-context
// two IRB interfaces with anycast-gw configuration is added
interface irb1 {
    subinterface 1 {
        ipv4 {
            admin-state enable
            address 20.0.0.1/24 {
                anycast-gw true
                primary
            }
            arp {
                learn-unsolicited true
                evpn {
                    advertise dynamic {
  // for ARP synchronization across MH leaf nodes
                    }
                }
            }
        }
        anycast-gw {
        }
    }
}
interface irb2 {
    subinterface 1 {
        ipv4 {
            admin-state enable
            address 30.0.0.1/24 {
                anycast-gw true
                primary
            }
            arp {
                learn-unsolicited true
                evpn {
                    advertise dynamic {
                    }
                }
            }
```


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

```
        }
        anycast-gw {
        }
    }
}
// lags associated with the ethernet-segment.
 In this case static, but they can be lacp based too.
interface lag1 {
    admin-state enable
    vlan-tagging true
    subinterface 1 {
        vlan {
            encap {
                single-tagged {
                    vlan-id 20
                }
            }
        }
    }
    subinterface 2 {
        vlan {
            encap {
                single-tagged {
                    vlan-id 30
                }
            }
        }
    }
    lag {
        lag-type static
    }
}
// ES configuration
system {
    network-instance {
        protocols {
            evpn {
                ethernet-segments {
                    bgp-instance 1 {
                        ethernet-segment ES-4 {
                            admin-state enable
                            esi 00:44:44:44:44:44:44:00:00:04
                            interface lag1
                        }
                    }
                }
            }
        }
    }
}
// MAC-VRFs
network-instance MAC-VRF-X {
    type mac-vrf
    admin-state enable
    interface irb1.1 {
    }
    interface lag1.1 {
    }
    vxlan-interface vxlan1.20 {
    }
    protocols {
        bgp-evpn {
            bgp-instance 1 {
                admin-state enable
```


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

```
                vxlan-interface vxlan1.20
                evi 20
            }
        }
        bgp-vpn {
            bgp-instance 1 {
            }
        }
    }
}
network-instance MAC-VRF-Y {
    type mac-vrf
    admin-state enable
    interface irb2.1 {
    }
    interface lag1.2 {
    }
    vxlan-interface vxlan1.30 {
    }
    protocols {
        bgp-evpn {
            bgp-instance 1 {
                admin-state enable
                vxlan-interface vxlan1.30
                evi 30
            }
        }
        bgp-vpn {
            bgp-instance 1 {
            }
        }
    }
}
// default network-instance configuration
network-instance default {
    type default
    admin-state enable
    description "Default network instance"
    router-id 1.1.1.1
    interface ethernet-1/1.1 {
    }
    interface ethernet-1/20.1 {
    }
    interface irb1.1 {
    }
    interface irb2.1 {
    }
    interface system0.0 {
    }
    protocols {
        bgp {
            admin-state enable
            afi-safi ipv4-unicast {
                admin-state enable
            }
            autonomous-system 1234
            router-id 1.1.1.1
            group eBGP-spines {
                admin-state enable
                export-policy export-all
                peer-as 4567
                afi-safi ipv4-unicast {
                    admin-state enable
                }
```


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

```
            }
            group evpn-mh {
                admin-state enable
                export-policy export-all
                peer-as 1234
                afi-safi evpn {
                    admin-state enable
                }
            }
            neighbor 2.2.2.2 {
                admin-state enable
                peer-group evpn-mh
            }
            neighbor 10.1.4.4 {
                admin-state enable
                peer-group eBGP-spines
            }

        }
    }
}
// vxlan interfaces for the MH configuration
tunnel-interface vxlan1 {
    vxlan-interface 20 {
        type bridged
        ingress {
            vni 20
        }
    }
    vxlan-interface 30 {
        type bridged
        ingress {
            vni 30
        }
    }
}
```

Leaf 1 in Figure 65: Hierarchical active-active connectivity example has the following configuration. The configuration for Leaf 2 would be equivalent.

```
--{ [FACTORY] +* candidate shared default }--[  ]--
A:Leaf-1# info with-context
// lags associated with the ethernet-segments.
 In this case static, but they can be lacp based too.
interface lag1 {
    admin-state enable
    vlan-tagging true
    subinterface 1 {
        vlan {
            encap {
                single-tagged {
                    vlan-id 20
                }
            }
        }
    }
    lag {
        lag-type static
    }
}
interface lag2 {
    admin-state enable
    vlan-tagging true
```


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

```
    subinterface 1 {
        vlan {
            encap {
                single-tagged {
                    vlan-id 30
                }
            }
        }
    }
    lag {
        lag-type static
    }
}
interface lag3 {
    admin-state enable
    vlan-tagging true
    subinterface 1 {
        vlan {
            encap {
                single-tagged {
                    vlan-id 20
                }
            }
        }
    }
    subinterface 2 {
        vlan {
            encap {
                single-tagged {
                    vlan-id 30
                }
            }
        }
    }
    lag {
        lag-type static
    }
}
// ES configuration
system {
    network-instance {
        protocols {
            evpn {
                ethernet-segments {
                    bgp-instance 1 {
                        ethernet-segment ES-1 {
                            admin-state enable
                            esi 00:11:11:11:11:11:11:00:00:01
                            interface lag1
                        }
                        ethernet-segment ES-2 {
                            admin-state enable
                            esi 00:22:22:22:22:22:22:00:00:02
                            interface lag2
                        }
                        ethernet-segment ES-3 {
                            admin-state enable
                            esi 00:33:33:33:33:33:33:00:00:03
                            interface lag3
                        }
                    }
                }
            }
        }
```


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

```
    }
}
// MAC-VRFs
network-instance MAC-VRF-X {
    type mac-vrf
    admin-state enable
    interface lag1.1 {
    }
    interface lag3.1 {
    }
    vxlan-interface vxlan1.20 {
    }
    protocols {
        bgp-evpn {
            bgp-instance 1 {
                admin-state enable
                vxlan-interface vxlan1.20
                evi 20
            }
        }
        bgp-vpn {
            bgp-instance 1 {
            }
        }
    }
}
network-instance MAC-VRF-Y {
    type mac-vrf
    admin-state enable
    interface lag2.1 {
    }
    interface lag3.1 {
    }
    vxlan-interface vxlan1.30 {
    }
    protocols {
        bgp-evpn {
            bgp-instance 1 {
                admin-state enable
                vxlan-interface vxlan1.30
                evi 30
            }
        }
        bgp-vpn {
            bgp-instance 1 {
            }
        }
    }
}
// default network-instance configuration
network-instance default {
    type default
    admin-state enable
    description "Default network instance"
    router-id 1.1.1.1
    interface ethernet-1/1.1 {
    }
    interface system0.0 {
    }
    protocols {
        bgp {
            admin-state enable
            afi-safi ipv4-unicast {
                    admin-state enable
```


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

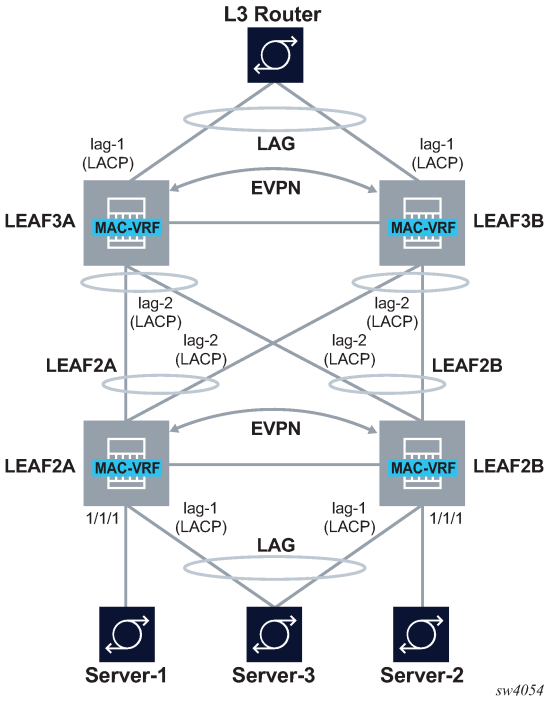
```
                }
            autonomous-system 1234
            router-id 1.1.1.1
            group evpn-mh {
                admin-state enable
                export-policy export-all
                peer-as 1234
                afi-safi evpn {
                    admin-state enable
                }
            }
            neighbor 2.2.2.2 {
                admin-state enable
                peer-group evpn-mh
            }
        }
    }
}
// vxlan interfaces for the MH configuration
tunnel-interface vxlan1 {
    vxlan-interface 20 {
        type bridged
        ingress {
            vni 20
        }
    }
    vxlan-interface 30 {
        type bridged
        ingress {
            vni 30
        }
    }
}
```

7.3 EVPN multihoming as standalone solution for MC-LAG

EVPN multihoming is not only used in overlay DCs, but also as a standalone solution for multihoming in Layer 2 access networks with no VXLAN. The following figure shows this usage.

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

Figure 66: EVPN multihoming as standalone MC-LAG solution



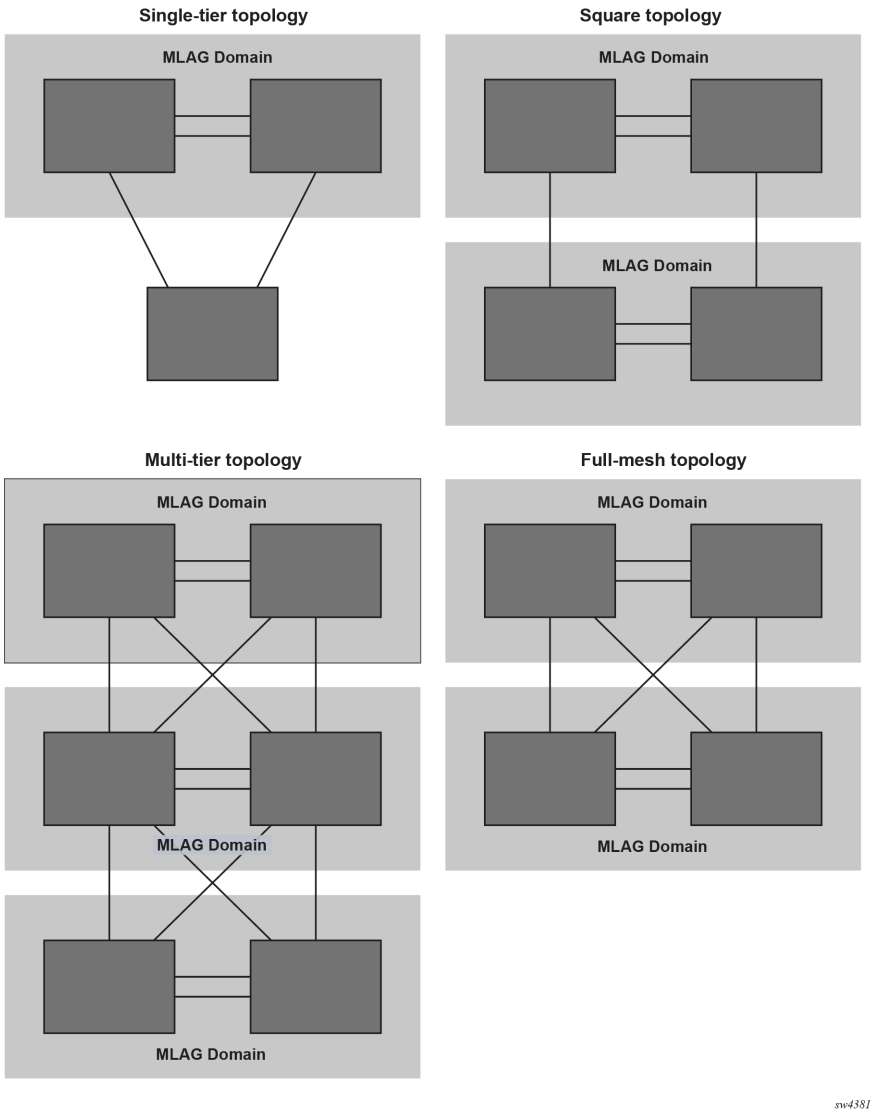
On the left side of the figure, EVPN multihoming is used as a standalone multihoming solution for leaf nodes connected via bridged subinterfaces.

Leafs of a layer do not use VXLAN to get connected to the higher layer. In this case, EVPN sessions are configured locally within each multihoming pair so that EVPN handles DF Election, split-horizon and synchronization of MAC and ARP entries. However, the leafs of different layers are not connected through any IP fabric, so no VXLAN or EVPN is needed end-to-end.

In this configuration, EVPN provides an alternative to MC-LAG solutions, being able to match all the topologies that other MC-LAG solutions support. These topologies include single-tier, multi-tier, square, or full-mesh/bow-tie topologies, as shown in the following figure. EVPN multihoming is supported in all of them as a replacement of MC-LAG.

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

Figure 67: MLAG topologies



7.3.1 Configuration for EVPN multihoming as standalone MC-LAG

LEAF2A in Figure 66: EVPN multihoming as standalone MC-LAG solution has the following configuration:

```
// lag1 connects LEAF2A to server-3
 and is associated to an all-active Ethernet Segment.
--{ candidate shared default }--[  ]--
A:LEAF2A# info with-context interface lag*
    interface lag1 {
        admin-state enable
        vlan-tagging true
        subinterface 10 {
            type bridged
```


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

```
            vlan {
                encap {
                    single-tagged {
                        vlan-id 10
                    }
                }
            }
        }
        lag {
            lag-type static
 // lag-type could also be lacp, in which case the
 system-id/key must match on lag1 of LEAF2B
            member-speed 10G
        }
    }
// lag2 connects LEAF2A to LEAF3A and LEAF3B.
 This LAG is an access LAG (does not carry vxlan) associated to
 an all-active Ethernet Segment
    interface lag2 {
        admin-state enable
        vlan-tagging true
        subinterface 10 {
            type bridged
            vlan {
                encap {
                    single-tagged {
                        vlan-id 10
                    }
                }
            }
        }
        lag {
            lag-type static
 // lag-type could also be lacp, in which case the
 system-id/key must match on lag2 of LEAF2B
            member-speed 10G
        }
    }
// A vxlan-interface is created for the inter-chassis traffic
--{ candidate shared default }--[  ]-
A:LEAF2A# info with-context tunnel-interface vxlan1 vxlan-interface 10
    tunnel-interface vxlan1 {
        vxlan-interface 10 {
            type bridged
            ingress {
                vni 10
            }
        }
    }
// the Ethernet Segments associated to lag1 and lag2
--{ candidate shared default }--[  ]--
A:LEAF2A# info with-context system network-instance protocols evpn
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment ES-leaf1-leaf2.CE1 {
                                admin-state enable
                                esi 00:12:12:12:12:12:12:00:00:01
                                interface lag1
                            }
                            ethernet-segment ES-leaf1-leaf2.Spines {
```


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

```
                                admin-state enable
                                esi 00:12:12:12:12:12:12:00:00:02
                                interface lag2
                            }
                        }
                    }
                }
            }
        }
// the mac-vrf uses lag sub-interfaces for the connectivity to rest of
 the leaf nodes and servers, and a vxlan-subinterface for the connectivity
 to LEAF2B
--{ candidate shared default }--[  ]--
A:LEAF2A# info with-context network-instance Blue-MAC-VRF-10
    network-instance Blue-MAC-VRF-10 {
        type mac-vrf
        interface ethernet-1/1.10 {
            // this is connected to a single-homed access server-1
        }
        interface lag1.10 {
            // access lag - multi-homed to access server-3
        }
        interface lag2.10 {
            // multi-homed to spines
        }
        vxlan-interface vxlan1.10 {
            // vxlan-interface used for inter-chassis connectivity only
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.10
                    evi 10
                }
            }
            bgp-vpn {
            }
        }
    }
```

LEAF2A in Figure 66: EVPN multihoming as standalone MC-LAG solution has the following configuration:

```
// lag1 connects LEAF2B to server-3 and is associated to
 an all-active Ethernet Segment
--{ candidate shared default }--[  ]--
A:LEAF2B# info with-context interface lag*
    interface lag1 {
        admin-state enable
        vlan-tagging true
        subinterface 10 {
            type bridged
            vlan {
                encap {
                    single-tagged {
                        vlan-id 10
                    }
                }
            }
        }
        lag {
            lag-type static
```


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

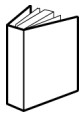
```
 // lag-type could also be lacp, in which case the
 system-id/key must match on lag1 of LEAF2A
            member-speed 10G
        }
    }
// lag2 connects LEAF2B to LEAF3A and LEAF3B.
 This LAG is an access LAG (does not carry vxlan) associated to
 an all-active Ethernet Segment
    interface lag2 {
        admin-state enable
        vlan-tagging true
        subinterface 10 {
            type bridged
            vlan {
                encap {
                    single-tagged {
                        vlan-id 10
                    }
                }
            }
        }
        lag {
            lag-type static
 // lag-type could also be lacp, in which case the
 system-id/key must match on lag2 of LEAF2B
            member-speed 10G
        }
    }
// A vxlan-interface is created for the inter-chassis traffic
--{ candidate shared default }--[  ]--
A:LEAF2B# info with-context tunnel-interface vxlan1 vxlan-interface 10
    tunnel-interface vxlan1 {
        vxlan-interface 10 {
            type bridged
            ingress {
                vni 10
            }
        }
    }
// the Ethernet Segments associated to lag1 and lag2
--{ candidate shared default }--[  ]--
A:LEAF2B# info with-context system network-instance protocols evpn
    system {
        network-instance {
            protocols {
                evpn {
                    ethernet-segments {
                        bgp-instance 1 {
                            ethernet-segment ES-leaf1-leaf2.CE1 {
                                admin-state enable
                                esi 00:12:12:12:12:12:12:00:00:01
                                interface lag1
                            }
                            ethernet-segment ES-leaf1-leaf2.Spines {
                                admin-state enable
                                esi 00:12:12:12:12:12:12:00:00:02
                                interface lag2
                            }
                        }
                    }
                }
            }
        }
    }
```


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

```
// the mac-vrf uses lag sub-interfaces for the connectivity to rest of the
 leaf nodes and servers, and a vxlan-subinterface for the connectivity
 to LEAF2B
--{ candidate shared default }--[  ]--
A:LEAF2B# info with-context network-instance Blue-MAC-VRF-10
    network-instance Blue-MAC-VRF-10 {
        type mac-vrf
        interface ethernet-1/2.10 {
            !!! this is connected to a single-homed access server-2
        }
        interface lag1.10 {
            !!! access lag - multi-homed to access server-3
        }
        interface lag2.10 {
            !!! multi-homed to spines
        }
        vxlan-interface vxlan1.10 {
            !!! vxlan-interface used for inter-chassis connectivity only
        }
        protocols {
            bgp-evpn {
                bgp-instance 1 {
                    admin-state enable
                    vxlan-interface vxlan1.10
                    evi 10
                }
            }
            bgp-vpn {
            }
        }
    }
```


© 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