Nokia Service Router Linux
7220 Interconnect Router
7250 Interconnect Router
7730 Service Interconnect Router

Release 26.3

OAM and Diagnostics Guide

3HE 22245 AAAA TQZZA
Edition: 01
March 2026

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.

Table of contents

1 About this guide

This guide describes how to configure features such as mirroring and sFlow, and how to use the Operations, Administration, and Maintenance (OAM) and diagnostics tools with the Nokia Service Router Linux (SR Linux).

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 that will be released in later maintenance loads. See the *SR Linux Software Release Notes* for information about features supported in each load.

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

1.1 Precautionary and information messages

The following are information symbols used in the documentation.

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

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

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

Note: Note provides additional operational information.

Tip: Tip provides suggestions for use or best practices.

1.2 Conventions

The Nokia SR Linux documentation uses the following command conventions:

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

- 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
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
7250 IXR[1]	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-6e/10e (mixed system)	7250 IXR-6e (mixed system)[2], 7250 IXR-10e (mixed system)[2]
7730 SXR	7730 SXR-1-32D, 7730 SXR-1d-32D, 7730 SXR-1x-44S

1 References to the 7250 IXR platform group may be appended with (including mixed systems) or (excluding mixed systems) to indicate mixed system support.

2 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.

2 What's new

Table 2: What's new in Release 26.3.1

Topic	Location
Supports 7250 IXR Gen 3	Mirroring Service Activation Testhead (SAT)
OAM support in Segment Routing IPv6 (SRv6)	OAM support in Segment Routing IPv6 (SRv6)
Supports SAT and PLQ interactions	Service Activation Testhead and Packet Link Qualification
Supports sFlow sample sequence number Supports sFlow single source ID per interface	sFlow
Supports IPFIX	IPFIX

3 Mirroring

Mirroring consists of two main components: a mirror source and a mirror destination. It copies IPv4 and IPv6 packets. Mirror sources such as an interface (port), a subinterface (VLAN), or traffic matching an ACL entry are specified and copied exactly as they are seen on the wire. These packets are then sent to a specified mirror destination, such as a locally attached traffic analyzer or a remote destination via tunneling.

By default, mirrored packets include both Ethernet headers and IPv4/IPv6 headers. Traffic from multiple sources can be mirrored to a single destination. However, mirroring to multiple destinations is not supported.

3.1 Mirror sources

The source for mirrored traffic can be an interface, subinterface, or an ACL filter.

Interfaces and subinterfaces

A mirror source can be an interface, including all subinterfaces within that interface. The source can be:

- A single interface (for example, **interface ethernet-1/1**)
- A LAG (for example, **interface lag1**)

Either a LAG member or the LAG port can be configured as a mirror source.

When a LAG port is configured as a mirror source, mirroring captures all packets from all interfaces that make up the LAG. In this case, the system prevents these interfaces from being configured as additional mirror sources.

Conversely, if an interface is configured as a mirror source and is also a member of a LAG, the system rejects any attempt to configure the LAG port as another mirror source.

The mirror source can also be a specific VLAN, represented by a subinterface with VLAN tagging enabled (for example, **interface ethernet-1/1.1** or **interface lag1.1**).

You can configure mirroring for traffic in a specific direction, ingress only, egress only, or for bidirectional traffic (both ingress and egress).

If system resources are exhausted, the system may stop the mirroring process, causing the mirror source operational state to go down. If mirroring stops working, check the operational state of the mirror sources. The operational state and operational down reason help identify the cause of the issue for a specific mirror source.

Note:
On 7250 IXR Gen 3 platforms, an interface cannot simultaneously support sFlow egress sampling and act as a source for egress mirroring. If both are configured, the system rejects the configuration.

To enable sFlow egress sampling on an interface, you must first remove the interface as an egress mirror source. Conversely, to configure an interface for egress mirroring, you must first remove the sFlow configuration from that interface.

ACL filters

A mirror source can also be an IPv4 or IPv6 ACL filter applied to one or more interfaces or subinterfaces. Traffic that matches entries in an ingress ACL filter can be mirrored to the destination, regardless of whether the ACL action is accept or drop.

Supported platforms

The following table lists hardware platform support for each mirror source.

Table 3: Hardware applicability (source mirroring)

Source	7220 IXR-D2 7220 IXR-D3	7220 IXR-D2L 7220 IXR-D3L	7220 IXR-D4 7220 IXR-D5	7250 IXR-6e 7250 IXR-10e 7250 IXR Gen 2c+ 7250 IXR Gen 3	7250 IXR-X1b 7250 IXR-X3b	7730 SXR
Interface (ingress)	Yes	Yes	Yes	Yes	Yes	Yes
Interface (egress)	Yes	Yes	Yes	Yes	Yes	Yes
Subinterface (ingress)	Yes	Yes	Yes	Yes	Yes	Yes
Subinterface (egress)	Yes	Yes	No	Yes	Yes	Yes
ACL filter (ingress)	Yes	Yes	Yes	Yes	Yes	No
ACL filter (egress)	No	No	No	No	No	No

Note: The following considerations apply:

- On 7250 IXR systems, the subinterface used as mirror source cannot be of type bridged.
- On 7250 IXR systems, the LAG member port cannot be mirrored. Only the entire LAG can be mirrored.

3.2 Mirror destinations

Traffic from the mirror source can be copied to a local destination (local mirroring) or tunnel to a remote destination (remote mirroring).

3.2.1 Local mirroring

In a local mirroring configuration, both the mirror source and mirror destination reside on the same SR Linux node, as shown in the following figure.

In this configuration, the local destination is a Switched Port Analyzer (SPAN).

Figure 1: Local mirroring

For local mirroring, the following hardware types are supported:

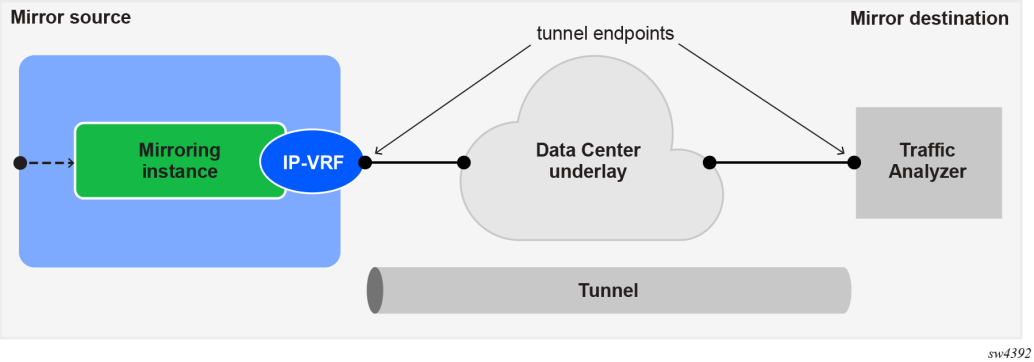
- 7220 IXR-D2
- 7220 IXR-D3
- 7220 IXR-D2L
- 7220 IXR-D3L
- 7220 IXR-D4
- 7220 IXR-D5
- 7250 IXR-6e
- 7250 IXR-10e
- 7250 IXR-X1b
- 7250 IXR-X3b
- 7250 IXR Gen 2c+
- 7250 IXR Gen 3
- 7730 SXR

3.2.2 Remote mirroring

In a remote mirroring setup, the mirror source and destination are located on different nodes. As shown in the following figure, the SR Linux node acts as the mirror source, and the mirrored packets are encapsulated into a tunnel toward the mirror destination.

Tunnel endpoints are defined within a specific network-instance, where the local tunnel endpoint IP address must either a loopback subinterface address or any subinterface address within that network-instance.

Figure 2: Remote mirroring



The following table summarizes the mirror destination types supported for various platforms. The sections that follow provide more details about each of the remote mirroring tunneling mechanisms.

Table 4: Hardware applicability (destination mirroring - remote)

Destination	7220 IXR-D2 7220 IXR-D3	7220 IXR-D2L 7220 IXR-D3L	7220 IXR-D4 7220 IXR-D5	7250 IXR-6e 7250 IXR-10e 7250 IXR Gen 2c+ 7250 IXR Gen 3	7250 IXR-X1b 7250 IXR-X3b	7730 SXR
Underlay destination (GRE +ERSPAN II) - IPv4 (ingress and egress	No	No	No	Yes	Yes	No
Underlay destination (GRE +ERSPAN II) -	No	No	No	Yes	Yes	No

Destination	7220 IXR-D2 7220 IXR-D3	7220 IXR-D2L 7220 IXR-D3L	7220 IXR-D4 7220 IXR-D5	7250 IXR-6e 7250 IXR-10e 7250 IXR Gen 2c+ 7250 IXR Gen 3	7250 IXR-X1b 7250 IXR-X3b	7730 SXR
IPv6 (ingress and egress)						
Underlay destination (transparent Ethernet bridging) - IPv4 (ingress and egress)	Yes	Yes	Yes	No	No	No
Underlay destination (transparent Ethernet bridging) - IPv6 (ingress-direction mirroring)	Yes	Yes	Yes	No	No	No
Underlay destination (transparent Ethernet bridging) - IPv6 (egress-direction mirroring)	No	No	Yes	No	No	No
IPv4 GRE pseudowire	No	No	No	No	No	Yes
MPLS pseudowire	No	No	No	No	No	Yes

Note: MPLS pseudowires are set up without a control word.

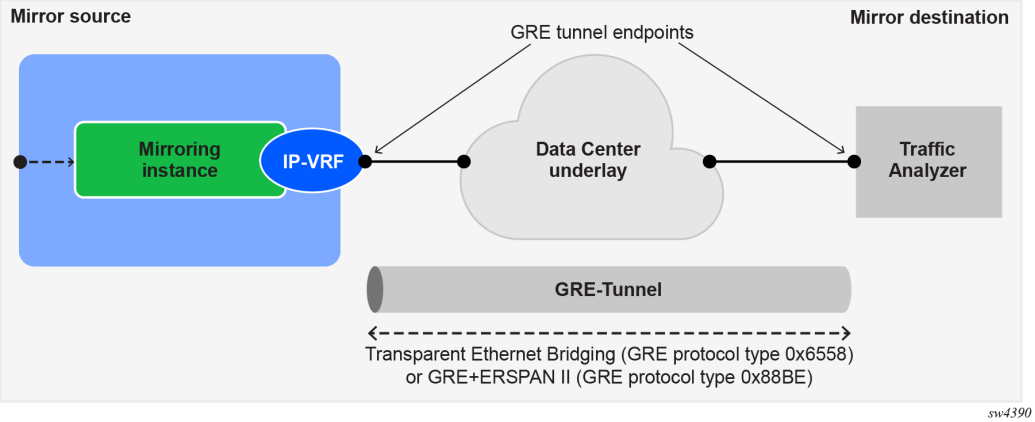
3.2.2.1 Mirroring utilizing an underlay IP infrastructure

For 7220 IXR-D3, 7220 IXR-D4, and 7220 IXR-D5 devices, the mirrored packets including the Ethernet headers are tunneled to the remote mirror destination with a GRE header using transparent Ethernet bridging (GRE protocol type 0x6558).

For 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, and 7250 IXR-X3b devices, the mirrored packets including the Ethernet headers are tunneled to the remote mirror destination. The mirrored packet is first encapsulated with an ERSPAN header and then an outer IP-GRE header using Ethernet bridging (GRE protocol type 0x88BE).

The following figure shows a mirroring-to-underlay configuration.

Figure 3: Mirroring to underlay



Consider the following when you configure remote mirroring:

- The system does not place any restrictions on the configuration of the **tunnel-end-point** destination or source IP address.
- For an ERSPAN to be functional:
  - the source IP address configured under **tunnel-end-point** address must be a local interface IP address. This is typically a loopback address on the system
  - there must be a route entry in the routing table destined for the **tunnel-end-point** destination IP address.

3.2.2.2 Mirroring utilizing a MPLS pseudowire

Note: This feature is currently supported exclusively on the 7730 SXR platforms.

Most MPLS-capable routers at the remote end can remove the encapsulation and forward the mirrored packet to an analyzer. If you are using IP-GRE, you can also specify an analyzer as the IP destination. In this case, the analyzer receives the full mirrored packet along with the MPLS service label encapsulation.

The 7730 SXR platforms support the following types of transport tunnels for mirroring the traffic:

- GRE
- SR-ISIS
- Colored TE policy
- Uncolored TE policy

The tunnel selection is automatic and based on dynamic resolution. The following operational state parameters display the selected tunnel type:

- **operational-tunnel-type**: Indicates the resolved tunnel type used for the mirror destination.
- **operational-tunnel-id**: Identifies the specific tunnel instance in use.
- **oper-down-reason**: Specifies the reason why the mirror session is in an oper-down state.

3.2.3 Mirror slicing

Note: This feature is currently supported exclusively on the 7730 SXR platforms.

Slicing is a function that only mirrors a specified packet length of the original packet. This is useful to monitor network usage without having to copy the full packet. Slicing can reduce the mirrored packet to a size that the destination packet decoding equipment can handle. It also allows the conservation of mirroring bandwidth consumption by limiting the size of the stream of packets through the router and the core network.

When a mirror **slice-size** is defined, it truncates the mirrored frame to the defined size. For example, if the slice size is configured as 256 bytes, only the first 256 bytes of the frame are transmitted to the mirror destination. The original frame is not affected by the truncation. Mirrored frames can be larger then the specified slice size if the mirror destination adds encapsulations for a remote decoder equipment.

The transmission of a sliced or a non-sliced frame is also dependent on the mirror destination path MTU or the mirror destination MTU. Packets with MTU larger than the mirroring destination are discarded.

3.3 Configuring mirroring

To configure mirroring, you configure a mirroring-instance, which specifies the source and destination for the mirrored traffic. Multiple mirror sources can have a single destination, although traffic from a specific source cannot be mirrored to multiple destinations. Only one mirror destination can be configured per mirroring-instance. A mirror destination cannot be reused in multiple mirroring instances.

Within a mirroring-instance, if an interface is configured as mirror source, a subinterface within that interface cannot be added as another mirror source. If a LAG is defined as mirror destination, only the first 8 members of the LAG carry mirrored traffic. Note that on 7220 IXR-D4 and 7220 IXR-D5 platforms, a mirror destination port cannot be a LAG.

Mirrored traffic is considered Best Effort (BE) Forwarding Class.

3.3.1 Configuring mirroring sources

Procedure

To configure mirroring, you specify the source and destination for mirrored traffic within a mirroring-instance. The source in a mirroring-instance can be traffic on a specified interface, subinterface, or LAG, or can be packets matching an ACL entry.

Example: interface source

The following example shows a mirroring-instance configuration with an interface as the source for mirrored traffic:

```
--{ + candidate shared default }--[  ]--
# info with-context system mirroring
    system {
        mirroring {
            mirroring-instance 1 {
                admin-state enable
                mirror-source {
                    interface ethernet-1/5 {
                        direction ingress-egress
                    }
                }
            }
        }
    }
```

Example: ACL source

The following example configures an ACL with an entry that matches TCP packets and applies the ACL to a subinterface. A mirroring-instance is configured that uses packets matching the ACL as the source for mirrored traffic.

```
--{ + candidate shared default }--[  ]--
# info with-context acl acl-filter ip_tcp type ipv4
    acl {
        acl-filter ip_tcp type ipv4 {
            entry 1000 {
                description Match_TCP_Protocol
                match {
                    ipv4 {
                        protocol tcp
                    }
                }
                action {
                    accept {
                    }
                }
            }
        }
    }
```

```
--{ + candidate shared default }--[  ]--
# info with-context acl interface ethernet-1/1.1
    acl {
        interface ethernet-1/1.1 {
            interface-ref {
                interface ethernet-1/1
                subinterface 1
            }
            input {
                acl-filter ip_tcp type ipv4 {
                }
            }
        }
```

```
    }
```

```
--{ + candidate shared default }--[  ]--
# info with-context system mirroring
    system {
        mirroring {
            mirroring-instance 1 {
                admin-state enable
                mirror-source {
                    interface ethernet-1/5 {
                        direction ingress-egress
                    }
                    acl {
                        acl-filter ip_tcp type ipv4 {
                            entry 1000 {
                            }
                        }
                    }
                }
            }
        }
    }
```

3.3.2 Configuring mirroring destinations

Procedure

In a mirroring-instance, you specify the destination for the mirrored traffic.

The mirroring destination can be a local destination residing on the same SR Linux node as the mirroring source. See Configuring a local mirroring destination for an example of a local mirroring destination configuration.

The mirroring destination can be a remote destination where the mirrored traffic is sent via a tunnel. See Configuring a remote mirroring destination using underlay for an example of a remote mirroring destination configuration.

The 7250 IXR platforms (7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, and 7250 IXR-X3b) support mirroring of packets including the original L2 header. The mirrored packets are then encapsulated with the GRE header and the ERSPAN II header of type 0x88BE.

On 7730 SXR platforms, mirrored packets are encapsulated as follows:

- if the operational-tunnel-type is GRE, the packet includes a GRE header with Ethernet type 0x8847 followed by the MPLS service label configured in the mirroring context
- if the operation-tunnel-type is sr-te or sr-isis, the Ethernet Type field is also set to 0x8847, indicating an MPLS frame with the service label configured in the mirroring context placed at the bottom of the label stack.

See Configuring a remote mirroring destination in 7730 SXR platforms

Example: Configuring a local mirroring destination

The following example configures a subinterface to be a local mirror destination.

```
--{ + candidate shared default }--[  ]--
# info with-context interface ethernet-1/4 subinterface 1
    interface ethernet-1/4 {
        subinterface 1 {
```

```
            type local-mirror-dest
            admin-state enable
            vlan {
                encap {
                    single-tagged {
                        vlan-id 1127
                    }
                }
            }
            local-mirror-destination {
                admin-state enable
            }
        }
    }
```

The following example configures a mirroring instance and specifies the local mirror destination.

```
--{ + candidate shared default }--[  ]--
# info with-context system mirroring mirroring-instance local
    system {
        mirroring {
            mirroring-instance local {
                admin-state enable
                mirror-destination {
                    local ethernet-1/4.1
                }
            }
        }
    }
```

Example: Configuring a remote mirroring destination using underlay

The following example configures a mirroring instance and specifies the mirrored traffic be encapsulated into a tunnel within a network-instance. The mirrored traffic is encapsulated with a GRE header and a ERSPAN header and is tunneled to the remote destination.

Note:
For 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, and 7250 IXR-X3b, the remote encapsulation is L3oGRE. For all other platforms, the encapsulation is L2oGRE.

```
--{ + candidate shared default }--[  ]--
# info with-context system mirroring
    system {
        mirroring {
            mirroring-instance test {
                admin-state enable
                mirror-source {
                    interface ethernet-1/1 {
                        direction ingress-egress
                    }
                }
                mirror-destination {
                    remote {
                        encap l2ogre
                        network-instance IPVRF-1
                        tunnel-end-points {
                            source-address 192.168.1.53
                            destination-address 192.168.1.153
                        }
                    }
                }
```

```
            }
        }
    }
```

Example: Configuring a remote mirroring destination in 7730 SXR platforms

The following example configures a remote mirroring destination in 7730 SXR platforms. In this example, the original packet along with its Layer 2 header is mirrored. The encapsulation used is MPLS over GRE. The tunnel type supported includes IP-GRE, SR-ISIS, and SR-TE).

```
--{ + candidate shared default }--[  ]--
# info with-context system mirroring mirroring-instance one
    system {
        mirroring {
            mirroring-instance one {
                mirror-source {
                    subinterface lag7.1 {
                        direction ingress-only
                    }
                }
                mirror-destination {
                    remote {
                        encap mpls
                        network-instance green_default
                        tunnel-end-points {
                            source-address 1.1.1.1
                            destination-address 32.1.1.5
                            admin-state enable
                            service-label 145
                            allowed-tunnel-types [
                                sr-isis
                                te-policy-sr-mpls-uncolored
                            ]
                        }
                    }
                }
            }
        }
    }
```

3.3.3 Configuring mirror slice size

Procedure

Note: This feature is currently supported exclusively on the 7730 SXR platforms.

To configure a mirror slice size, use the **system mirroring mirroring-instance mirror-destination slice-size** command.

Example: Configure a mirror slice size

The following example configures a slice size for a mirror destination.

```
--{ + candidate shared default }--[  ]--
# info with-context system mirroring mirroring-instance test
    system {
        mirroring {
```

```
            mirroring-instance test {
                admin-state enable
                mirror-source {
                    interface ethernet-1/1 {
                        direction ingress-egress
                    }
                }
                mirror-destination {
                    slice-size 256
                    remote {
                        encap mpls
                        network-instance IPVRF-1
                        tunnel-end-points {
                            source-address 192.168.1.53
                            destination-address 192.168.1.153
                            service-label 16
                            allowed-tunnel-types [
                                gre
                            ]
                        }
                    }
                }
            }
        }
    }
```

3.4 Displaying mirroring information

Procedure

Use the **info from state** command to display mirroring configuration information.

Example: Displaying mirroring configuration information in 7250 IXR devices

```
--{ * candidate shared default }--[  ]--
# info from state with-context system mirroring mirroring-instance 2
    system {
        mirroring {
            mirroring-instance 2 {
                admin-state enable
                oper-state down
                oper-down-reason local-mirror-subif-down
                mirror-source {
                    interface lag1 {
                        direction ingress-egress
                    }
                }
                mirror-destination {
                    local lag25.1
                }
            }
        }
    }
```

Example: Displaying mirroring configuration information in 7730 SXR devices

```
--{ + candidate shared default }--[  ]--
# info from state with-context system mirroring
    mirroring-instance m1 {
```

```
        admin-state enable
        oper-state up
        mirror-source {
            interface ethernet-1/11 {
                direction ingress-only
            }
        }
        mirror-destination {
            slice-size 0
            remote {
                encap mpls
                network-instance base
                tunnel-end-points {
                    source-address 1.1.1.2
                    destination-address 1.1.1.3
                    admin-state enable
                    service-label 234
                    oper-state up
                    operational-tunnel-type sr-isis
                    operational-tunnel-id 23000
                    allowed-tunnel-types [
                        sr-isis
                        te-policy-sr-mpls-colored
                    ]
                }
            }
        }
    }
```

3.5 Displaying mirroring statistics

Procedure

On 7220 IXR-D2, 7220 IXR-D3, and 7730 SXR platforms, you can display the statistics per mirror destination interface using the **info from state interface statistics** command. Filter **out-mirrored-packets** and the **out-mirror-octets** fields. See Mirroring statistics on 7220 IXR-D2, 7220 IXR-D3, and 7730 SXR platforms for an example of displaying 7220 IXR-D2, 7220 IXR-D3, and 7730 SXR mirroring statistics.

On 7220 IXR-D4, 7220 IXR-D5, 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b. and 7250 IXR-X3b platforms, mirror destination statistics are not supported per-interface; it is only possible to display per-mirror-destination statistics. The statistics show the number of packets sent to the mirror destination. See Mirroring statistics on 7250 IXR-6e, 7250 IXR-10e platforms for an example of displaying 7250 IXR-6e, 7250 IXR-10e mirroring statistics.

On 7220 IXR-D4 and 7220 IXR-D5 platforms, the statistics only include the number of packets mirrored in either the ingress or the egress direction. On 7250 IXR-6e, 7250 IXR-10e, and 7250 IXR-X3b platforms, the statistics include the number of packets in the ingress direction and the number of octets mirrored in either the ingress or the egress direction.

See Mirroring statistics on 7220 IXR-D5 platform for an example of displaying 7220 IXR-D5 mirroring statistics.
The octet count for ERSPAN includes the GRE header (not just the actual mirror packet). The interfaces that egress the mirrored packet must adjust the MTU size to accommodate that additional GRE header. If the MTU size is smaller than the GRE packet, the mirrored packet is dropped.

There are no packet drop statistics for mirror destinations. The statistics represent all packets that have been successfully mirrored and sent to the mirror destination. It is possible for mirrored packets to be

dropped because of over-congestion of multiple mirror sources to the same mirror destination. Mirrored packet drops can also occur because a mirror destination interface can be used for regular data traffic forwarding.

Example: Mirroring statistics on 7220 IXR-D2, 7220 IXR-D3, and 7730 SXR platforms

```
--{ running }--[  ]--
# info from state with-context interface ethernet-1/48 statistics | filter fields out-
mirror-octets out-mirror-packets
    interface ethernet-1/48 {
        statistics {
            out-mirror-octets 0
            out-mirror-packets 0
        }
```

Example: Mirroring statistics on 7250 IXR-6e, 7250 IXR-10e platforms

```
--{ running }--[  ]--
# info from state with-context system mirroring mirroring-instance ixia_one mirror-
destination statistics
    system {
        mirroring {
            mirroring-instance ixia_one {
                mirror-destination {
                    statistics {
                        ingress-mirrored-packets 7417657
                        ingress-mirrored-octets 10384702600
                        egress-mirrored-octets 0
                    }
                }
            }
        }
    }
```

Example: Mirroring statistics on 7220 IXR-D5 platform

```
--{ running }--[  ]--
# info from state with-context system mirroring mirroring-instance * mirror-destination
 statistics
    system {
        mirroring {
            mirroring-instance eight {
                mirror-destination {
                    statistics {
                        ingress-mirrored-packets 22135
                        egress-mirrored-packets 22132
                    }
                }
            }
            mirroring-instance five {
                mirror-destination {
                    statistics {
                        ingress-mirrored-packets 6353567
                        egress-mirrored-packets 0
                    }
                }
            }
        }
    }
```

4 OAM fault and performance tools and protocols

Effective network operation, administration, and maintenance (OAM) rely on specialized tools and protocols designed for fault detection, isolation, and performance measurement. Various OAM mechanisms ensure network reliability and efficiency across different layers and technologies. MPLS OAM, IP OAM, and Ethernet OAM play a crucial role in monitoring and managing network performance. Additionally, protocols such as Bidirectional Forwarding Detection (BFD), Micro-BFD, and Seamless Bidirectional Forwarding Detection (S-BFD) enhance network resilience by enabling rapid failure detection and recovery.

4.1 IP OAM tools and protocols

This section provides information about the IP OAM tools and protocols.

4.1.1 ICMP ping and trace

Note: This feature is supported on 7250 IXR and 7730 SXR platforms.

Internet Control Message Protocol (ICMP) is part of the IP suite as defined in RFC 792, Internet Control Message Protocol, for IPv4 and RFC 4443, Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. ICMP and ICMPv6 send and receive control and error messages used to manage the behavior of the TCP/IP stack. ICMP and ICMPv6 provide the following:

- debugging tools and error reporting mechanisms to assist in troubleshooting an IP network
- the ability to send and receive error and control messages to far-end IP entities

Ping

The **ping** command uses an echo request message to elicit an echo response from a host or gateway. The **ping6** command is the IPv6 version of the **ping** command. See Performing an ICMP ping for more information.

Traceroute

The traceroute command is used to trace the route that the packets take from the current system to the destination. It uses the time to live (TTL) parameter to elicit an ICMP time exceeded response from each gateway along the path to the host. The **traceroute6** command is the IPv6 version of the **traceroute** command. See Performing an ICMP trace for more information.

4.1.1.1 Performing an ICMP ping

Procedure

Use the **ping** (IPv4) or **ping6** (IPv6) command to contact an IP address. Use this command in any mode.

Example: ping for IPV4

```
--{ running }--[  ]--
# ping 192.168.1.1 network-instance default
Pinging 192.168.1.1 in srbase-default
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.027 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=0.032 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=64 time=0.030 ms
^C
--- 192.168.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 6165ms
rtt min/avg/max/mdev = 0.027/0.030/0.033/0.005 ms
```

4.1.1.2 Performing an ICMP trace

Procedure

To display the path a packet takes to a destination, use the **traceroute** (IPv4) or **traceroute6** (IPv6) command.

To trace the route using TCP SYN packets instead of UDP or ICMP echo packets, use the **tcptraceroute** command.

Example: traceroute for IPv4

```
--{ running }--[  ]--
# traceroute 1.1.1.1  network-instance mgmt
Using network instance srbase-mgmt
traceroute to 10.1.1.1 (10.1.1.1), 30 hops max, 60 byte packets
 1  172.18.18.1 (172.18.18.1)  1.268 ms  1.260 ms  1.256 ms
 2  172.21.40.1 (172.21.40.1)  1.253 ms  1.848 ms  1.851 ms
 3  172.22.35.230 (172.22.35.230)  1.835 ms  1.834 ms  1.828 ms
 4  66.201.62.1 (66.201.62.1)  3.222 ms  3.222 ms  3.216 ms
 5  66.201.34.17 (66.201.34.17)  5.474 ms  5.475 ms  5.480 ms
 6  * * *
 7  206.81.81.10 (206.81.81.10)  32.577 ms  32.542 ms  32.400 ms
 8  10.1.1.1 (10.1.1.1)  22.627 ms  22.637 ms  22.638 ms
```

4.1.2 TWAMP

Note: This feature is supported on 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, 7250 IXR-X3b, 7250 IXR Gen 3 and 7730 SXR platforms.

Two-Way Active Measurement Protocol (TWAMP) is a standards-based method to measure the IP performance between two devices including packet loss, delay, and jitter. TWAMP leverages the

methodology and architecture of One-Way Active Measurement Protocol (OWAMP) to define a method to measure two-way or round-trip metrics.

Components

The following are the four logical entities in TWAMP:

- control client: initiates the TWAMP control session and negotiates the test session parameters with the server
- server: responds to the control client request with the capacities that are supported and negotiates test session parameters
- session sender: transmits test packets to the session reflector
- session reflector: transmits a packet to the session sender in response to each packet it receives

These four functional elements are typically coupled together. The control client and session sender are implemented in one physical device which is referred to as the client. The server and session reflector are implemented in a second physical device which is referred to as the server. SR Linux supports the server and the session reflector.

See Configuring a TWAMP server for more information about steps to configure a TWAMP server.

Protocols

The following protocols are used in TWAMP sessions:

- TWAMP control protocol (TCP port 862): used to establish and manage control sessions between the control client and the server
- TWAMP test protocol (configurable UDP port): used to generate and send test traffic between the session sender and session reflector, and to measure network performance metrics like delay

Establishing a control session

The control client initiates a TCP connection and exchanges TWAMP control messages over this connection. The server accepts the TCP control session from the control client and responds with a server greeting message. This greeting includes the modes that are supported by the server. The modes are in the form of a bit mask. Each bit in the mask represents a functionality supported on the server.

SR Linux server mode support includes:

- unauthenticated server
- individual session control (mode bit 4)
- reflected octets (mode bit 5)
- symmetrical size test packet (mode bit 6)

To start testing, the control client communicates the test parameters to the server, requesting any of the modes that the server supports. If the server agrees to conduct the described tests, the test begins as soon as the control client sends a start sessions or start-n-session message.

Executing a test session

The session sender initiates the test session by sending a stream of UDP-based TWAMP test packets to the session reflector. The session reflector responds to each received packet with a UDP-response TWAMP test packet. The exchange of TWAMP test PDUs is referred to as a TWAMP test. The session sender calculates the various delay and loss metric based on the received TWAMP test PDUs . The

TWAMP test PDU does not achieve symmetrical packet size in both directions unless the frame is padded with a minimum of 27 bytes. The session sender is responsible for applying the required padding. After the frame is appropriately padded, the session reflector reduces the padding by the number of bytes needed to provide symmetry.

The control client can terminate individual test by sending the appropriate stop message, or terminate all tests for a control channel by terminating the TCP control channel.

TWAMP statistics

The following TWAMP statistics are available in SR Linux:

- system-level TWAMP statistics
- server statistics
- client connection statistics
- control connection statistics
- session reflector statistics

See Displaying TWAMP statistics for more information.

A clear command is available at the server network-instance level to clear all test session transmit and receive statistics and error counters. See Clearing TWAMP session statistics for more information.

4.1.2.1 Configuring a TWAMP server

Procedure

SR Linux supports TWAMP for the network instance type, default. You can configure the server and session reflector is under the **oam twamp** context.

Example: Configure TWAMP server

The following example configures a TWAMP server.

```
--{ + candidate shared default }--[  ]--
# info with-context oam twamp
    oam {
        twamp {
            server {
                network-instance default {
                    admin-state enable
                    servwait 900
                    control-packet-dscp 12
                    enforce-test-session-start-time true
                    client-connection 192.15.20.9/32 {
                        maximum-connections 32
                        maximum-sessions 32
                    }
                }
            }
        }
    }
```

4.1.2.2 Displaying TWAMP statistics

Procedure

To display system-level TWAMP statistics, use the **info from state oam twamp** command.

Example: Displaying TWAMP statistics

The following example displays TWAMP statistics.

```
--{ + candidate shared default }--[  ]--
# info from state with-context oam twamp
    oam {
        twamp {
            server {
                network-instance default {
                    admin-state enable
                    oper-state up
                    servwait 60
                    control-packet-dscp CS7
                    enforce-test-session-start-time true
                    maximum-connections 64
                    maximum-sessions 128
                    modes [
                        unauthenticated
                        individual-session-control
                        reflect-octets
                        symmetrical-size
                    ]
                    statistics {
                        test-sessions-active 1
                        test-sessions-completed 0
                        test-sessions-rejected 0
                        test-sessions-aborted 0
                        test-packets-received 2
                        test-packets-transmitted 2
                        control-connections-active 1
                        control-connections-rejected 0
                    }
                    client-connection 10.32.5.0/24 {
                        maximum-connections 64
                        maximum-sessions 128
                        statistics {
                            test-sessions-active 1
                            test-sessions-completed 0
                            test-sessions-rejected 0
                            test-sessions-aborted 0
                            test-packets-received 2
                            test-packets-transmitted 2
                            control-connections-active 1
                            control-connections-rejected 0
                        }
                    }
                    client-connection 10.11.1.0/24 {
                        maximum-connections 64
                        maximum-sessions 128
                        statistics {
                            test-sessions-active 0
                            test-sessions-completed 0
                            test-sessions-rejected 0
                            test-sessions-aborted 0
                            test-packets-received 0
```

```
                            test-packets-transmitted 0
                            control-connections-active 0
                            control-connections-rejected 0
                        }
                    }
                    client-connection 10.12.1.0/24 {
                        maximum-connections 32
                        maximum-sessions 32
                        statistics {
                            test-sessions-active 0
                            test-sessions-completed 0
                            test-sessions-rejected 0
                            test-sessions-aborted 0
                            test-packets-received 0
                            test-packets-transmitted 0
                            control-connections-active 0
                            control-connections-rejected 0
                        }
                    }
                    client-connection 2001:db8:101:1:1::/120 {
                        maximum-connections 64
                        maximum-sessions 128
                        statistics {
                            test-sessions-active 0
                            test-sessions-completed 0
                            test-sessions-rejected 0
                            test-sessions-aborted 0
                            test-packets-received 0
                            test-packets-transmitted 0
                            control-connections-active 0
                            control-connections-rejected 0
                        }
                    }
                    client-connection 2001:db8:101:1:1::/120 {
                        maximum-connections 32
                        maximum-sessions 32
                        statistics {
                            test-sessions-active 0
                            test-sessions-completed 0
                            test-sessions-rejected 0
                            test-sessions-aborted 0
                            test-packets-received 0
                            test-packets-transmitted 0
                            control-connections-active 0
                            control-connections-rejected 0
                        }
                    }
                    client-connection 2001:db8:101:1:1::/120 {
                        maximum-connections 64
                        maximum-sessions 128
                        statistics {
                            test-sessions-active 0
                            test-sessions-completed 0
                            test-sessions-rejected 0
                            test-sessions-aborted 0
                            test-packets-received 0
                            test-packets-transmitted 0
                            control-connections-active 0
                            control-connections-rejected 0
                        }
                    }
                    control-connection 10.32.5.0 client-tcp-port 58116 server-ip 10.20.1.3 server-tcp-
port 862 {
                        state active
```

```
                        control-packet-dscp 20
                        statistics {
                            test-sessions-active 1
                            test-sessions-completed 0
                            test-sessions-rejected 0
                            test-sessions-aborted 0
                            test-packets-received 2
                            test-packets-transmitted 2
                        }
                    }
                    session-reflector {
                        test-session 10.32.5.0 sender-udp-port 20100 reflector-ip 10.20.1.3 reflector-
udp-port 862 {
                            test-session-id 0A:14:01:03:EA:0B:06:CC:1A:36:F3:B2:A3:97:A2:55
                            parent-connection-client-ip 32.32.5.2
                            parent-connection-client-tcp-port 58116
                            parent-connection-server-ip 10.20.1.3
                            parent-connection-server-tcp-port 862
                            test-packet-dscp 0
                            last-sequence-number-transmitted 1
                            last-sequence-number-received 0
                            statistics {
                                test-packets-received 2
                                test-packets-transmitted 2
                            }
                        }
                    }
                }
            }
            statistics {
                dropped-connections {
                    tcp-connection-closed 0
                    tcp-connection-fatal-error 0
                    tcp-unexpected-event 0
                    message-send-error 0
                    memory-allocation-error 0
                    no-client-prefix-match 0
                    maximum-global-limit-exceed 0
                    maximum-prefix-limit-exceed 0
                    unspecified-mode 0
                    unsupported-mode 0
                    control-command-not-valid 0
                    incorrect-stop-session-count 0
                    connection-timeout 0
                    no-internal-resource 0
                    non-zero-sid-in-client-control-message 0
                    invalid-invalid-hmac 0
                }
                dropped-connection-states {
                    idle 0
                    setup-wait 0
                    started 0
                    active 0
                    process-started 0
                    process-stop 0
                    process-tw-session 0
                }
                rejected-session {
                    invalid-ip-address-version 0
                    non-local-ip-destination 0
                    bad-type-p 0
                    padding-too-big 0
                    non-zero-mbz-value 0
                    non-zero-session-sender-sid 0
```

```
                    timeout-too-large 0
                    maximum-global-session-exceed 0
                    maximum-prefix-session-exceed 0
                    client-source-ip-unreachable 0
                    udp-port-in-use 0
                    duplicate-session 0
                    no-internal-resource 0
                    refwait-timeout 0
                }
                dropped-test-packet {
                    incorrect-packet-size 0
                    incorrect-source-address 0
                    arrived-before-start-time 0
                    no-start-sessions 0
                    invalid-error-estimate 0
                    reply-error 0
                    invalid-server-octets 0
                    invalid-symmetric-mbz 0
                }
            }
        }
    }
```

4.1.2.3 Clearing TWAMP session statistics

Procedure

You can clear the TWAMP session statistics for each network instance using the **tools oam twamp server network-instance default clear** command.

Example: Clearing TWAMP session statistics

The following example clears TWAMP session statistics for each network instance.

```
--{ + candidate shared default }--[  ]--
# tools oam twamp server network-instance default clear
```

4.1.3 STAMP

Note: This feature is supported on 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, 7250 IXR-X3b, 7730 SXR and 7250 IXR Gen 3 platforms.

The Simple Two-Way Active Measurement Protocol (STAMP) defined in RFC 8762 is a standards-based method to measure the IP performance without the use of a control channel to negotiate test session parameters.

The STAMP test PDU allows for the collection of frame delay, frame delay range, inter-frame delay variation, and frame loss ratio. The RFC 8972 STAMP Optional Extensions specification maintains the existing structure of the STAMP test PDU but redefines existing fields and adds the capability to include TLVs. SR Linux supports RFC 8762 and the structural changes with TLV processing in the options draft RFC 8972.

STAMP operation

For each routed network instance, the STAMP session sender transmits STAMP test packets to the destination UDP port of the session reflector. The session reflector receives the packets, processes the STAMP test packet, and sends them back to the session sender. The session sender receives the reflected packets and uses the timestamps and sequence numbers to calculate delay and loss performance metrics. The session reflector supports a prefix list which filters based on IPv4 or IPv6 addressing. The reflector is stateful and uses the tuple SIP, DIP, SP, DP, and SSID to identify individual STAMP test sessions.

See Configuring STAMP session reflector for more information about how to configure a session reflector.

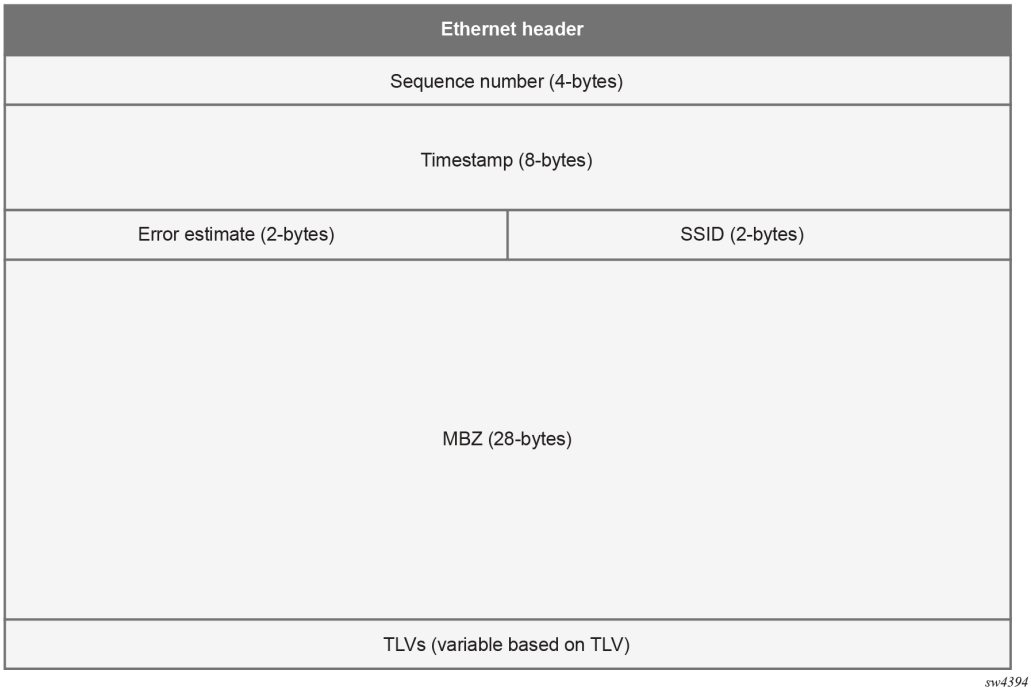
Session sender packet format

The session sender uses the tuple SIP, DIP, SP, and DP but uses the SSID to identify individual STAMP test sessions. The following table represents the STAMP test PDU (RFC8972) sent by the session sender encapsulated in a generic Ethernet header, with a description of the key protocol elements.

Table 5: Fields in a test request packet

Field	Description
Sequence Number	Packet sequence number generated based on the transmission sequence. For each new session, its value starts at 0 and is incremented by one with each transmitted packet.
Timestamp	Timestamp when a test packet is sent.
Error Estimate	Estimated error field. The format is as follows: • S bit is set to 0 regardless of time synchronization. • Z bit is set to 0 because the timestamp format is NTP. • Scale bits are set to 0. • Multiplier bits are non-zero.
SSID	Session Sender ID (SSID) automatically generated by the system.
MBZ	Must-Be-Zero (MBZ). The value must be 0. This field is used to ensure data packet symmetry between the session sender and session reflector.

Figure 4: Session sender packet format



Session reflector packet format

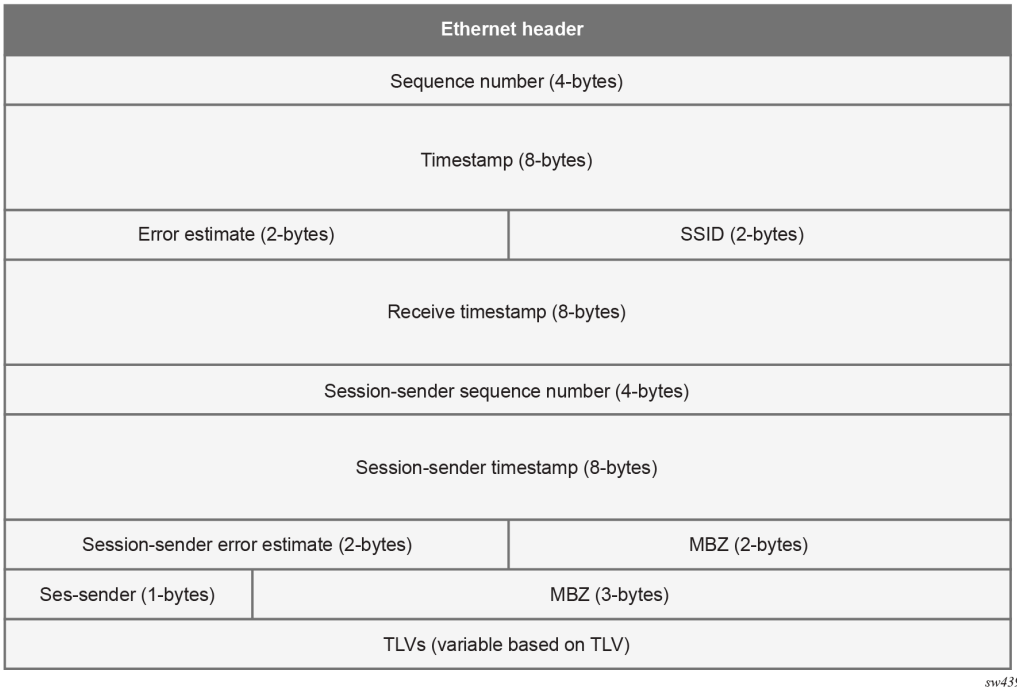
The following table represents the STAMP test PDU (RFC8972) sent by the session reflector encapsulated in a generic Ethernet header, with a description of the key protocol elements.

Table 6: Fields in a test response packet

Fields	Description
Sequence Number	Session reflector packet sequence number. For each new session, its value starts at 0 and is incremented by one with each transmitted packet.
Timestamp	Session reflector timestamp (T3)
Error Estimate	Session reflector estimated error field. The format is as follows: • S bit is set to 0 regardless of time synchronization. • Z bit is set to 0 because the timestamp format is NTP. • Scale bits are set to 0. • Multiplier bits are non-zero.
SSID	Session Sender ID (SSID) automatically generated by the system.
MBZ	Must-Be-Zero. The value must be 0. This field is used to ensure data packet symmetry between the Session-Sender and Session-Reflector.

Fields	Description
Receive Timestamp	Timestamp when a test packet is received on the session reflector (T2).
Session-Sender Sequence Number	Sequence Number field of the STAMP Test request packet.
Session-Sender Timestamp	Session sender timestamp field copied from the original the STAMP Test request packet (T1).
Session-Sender Error Estimate	Session sender Error Estimate field copied from the original the STAMP Test request packet.
Session-Sender TTL	Session sender TTL value copied from the original STAMP packet.

Figure 5: Session reflector packet format



Interoperability of STAMP and TWAMP Light session reflector

The following guidelines ensure that interoperability exists between STAMP and TWAMP Light by defining rules for packet processing based on packet size and content, particularly the 45th byte, to distinguish between the two protocols:

- UDP packets with a length less than 44 bytes are processed using TWAMP Light processing rules, which involves simple padding and symmetrical packet size handling.
- UDP packets with a length equal to 44 bytes are processed as STAMP packets.
- For UDP packets with a length equal to 45 bytes or more, the 45th byte is checked for the flags structure (100xxxxx).

  - If found, the packets are processed as STAMP packets.
  - If not found, the packet is assumed to be a TWAMP Light padded packet and processed accordingly. The TWAMP Light packet uses all zeros padding to avoid matching the 100xxxxx pattern by accident.
- Multiple TLVs in a STAMP test packet are parsed using the length field.
- If a TWAMP Light test packet mistakenly matches the 100xxxxx pattern at byte 45, the reflector attempts to parse the TLV. Failure to parse results in marking the byte as 110xxxxx and halting further STAMP TLV processing. However, the base STAMP packet continues to be processed.
- TWAMP Light packets arriving on a STAMP session reflector must use all zeros padding to avoid unintentional mismatching.

STAMP statistics

The following STAMP statistics are available in SR Linux:

- system-level session reflector statistics
- session reflector statistics for each network instance
- test session statistics

See Displaying STAMP statistics for more information.

4.1.3.1 Configuring STAMP session reflector

Procedure

To configure a STAMP session reflector for a network instance, use the **oam stamp** command and specify the network instance, IP address prefix, and the UDP port as shown in the example.

Example: Configuring STAMP session reflector

The following example configures a session reflector.

```
--{ + candidate shared default }--[  ]--
# info with-context oam stamp
    oam {
        stamp {
            session-reflector {
                network-instance default {
                    description test
                    admin-state enable
                    udp-port 862
                    ip-prefix 192.20.13.20/32 {
                    }
                }
            }
        }
    }
```

4.1.3.2 Displaying STAMP statistics

Procedure

To display system-level STAMP session reflector statistics, use the **info from state oam stamp** command.

Example: Displaying STAMP statistics

The following example displays STAMP statistics.

```
--{ + candidate shared default }--[  ]--
# info from state with-context oam stamp
  oam {
        stamp {
            session-reflector {
                inactivity-timer 900
                statistics {
                    test-frames-received 400
                    test-frames-sent 400
                    test-session-count 4
                    reflector-table-entries-full 0
                    packet-discards-on-reception 0
                    packet-discards-on-transmission 0
                    session-reflector-not-found 0
                    reflectors-configured 1
                    reflectors-operational 1
                    reflectors-not-operational 0
                }
                network-instance default {
                    admin-state enable
                    udp-port 862
                    oper-state up
                    ip-prefix  10.11.1.0/24 {
                    }
                    ip-prefix 10.10.11.0/24 {
                    }
                    ip-prefix 10.10.12.0/24 {
                    }
                    ip-prefix 10.10.14.0/24 {
                    }
                    ip-prefix 10.20.1.0/24 {
                    }
                    ip-prefix 10.12.1.0/24 {
                    }
                    ip-prefix 10.13.1.0/24 {
                    }
                    ip-prefix 10.14.1.0/24 {
                    }
                    ip-prefix 2001:db8:101:1:1/120 {
                    }
                    ip-prefix 2001:db8:102:1:1/120 {
                    }
                    ip-prefix 2001:db8:103:1:1/120 {
                    }
                    ip-prefix 2001:db8:104:1:1/120 {
                    }
                    ip-prefix 2001:db8:105:1:1/120 {
                    }
                    ip-prefix 2001:db8:106:1:1/120 {
                    }
                    ip-prefix 2001:db8:107:1:1/120 {
                    }
```

```
                    ip-prefix 2001:db8:108:1:1/120 {
                    }
                    statistics {
                        test-frames-received 400
                        test-frames-sent 400
                        test-sessions 4
                        prefix-match-failure 0
                        session-reflector-udp-port-registration-failure 0
                        malformed-packet 0
                        packet-discards-source-destination-equal 0
                    }
                    test-session-statistics 10.20.1.3 session-sender-udp 44000 session-reflector-ip
 11.20.1.2 session-reflector-udp 862 session-identifier 1736 {
                        last-sequence-number-received 99
                        last-sequence-number-transmitted 99
                        test-frames-received 100
                        test-frames-sent 100
                        malformed-tlv 0
                    }
                    test-session-statistics 10.10.3.3 session-sender-udp 44000 session-reflector-ip
 20.10.3.2 session-reflector-udp 862 session-identifier 1737 {
                        last-sequence-number-received 99
                        last-sequence-number-transmitted 99
                        test-frames-received 100
                        test-frames-sent 100
                        malformed-tlv 0
                    }
                    test-session-statistics 2001:db8:103:1:1 session-sender-udp 44000 session-
reflector-ip fc00::b14:102 session-reflector-udp 862 session-identifier 1738 {
                        last-sequence-number-received 99
                        last-sequence-number-transmitted 99
                        test-frames-received 100
                        test-frames-sent 100
                        malformed-tlv 0
                    }
                    test-session-statistics 2001:db8:104:1:1 session-sender-udp 44000 session-
reflector-ip fc00::140a:302 session-reflector-udp 862 session-identifier 1739 {
                        last-sequence-number-received 99
                        last-sequence-number-transmitted 99
                        test-frames-received 100
                        test-frames-sent 100
                        malformed-tlv 0
                    }
                }
```

4.2 MPLS OAM tools and protocols

Note: MPLS OAM tools are supported on 7730 SXR, 7250 IXR Gen 2c+, and 7250 IXR Gen 2 platforms.

This section provides information about the MPLS OAM tools and protocols.

4.2.1 LSP ping and trace

The LSP diagnostics include implementations of LSP ping and LSP trace based on RFC 8029, *Detecting Multiprotocol Label Switched (MPLS) Data Plane Failures*. LSP ping provides a mechanism to detect data

plane failures in MPLS LSPs. LSP ping and LSP trace are modeled after the ICMP echo request or reply used by ping and trace to detect and localize faults in IP networks.

For a specific LDP FEC, LSP ping verifies whether the packet reaches the egress label edge router (LER), while for LSP trace, the packet is sent to the control plane of each transit Label Switching Router (LSR) that performs various checks to see if it is intended to be a transit LSR for the path.

The downstream mapping TLV is used in LSP ping and LSP trace to provide a mechanism for the sender and responder nodes to exchange and validate interface and label stack information for each downstream hop in the path of an LDP FEC.

Note: UDP checksum is enforced for both LSP ping and LSP trace packets. Upon receiving these packets, the UDP checksum is validated. Packets with an invalid checksum are dropped. For IPv4 packets, a UDP checksum value of zero is permitted. However, for IPv6 packets, a checksum value of zero is considered invalid and such packets are dropped. Similarly, if the UDP checksum is non-zero but fails validation, the packet is also dropped.

See the following topics for more information about performing LSP ping and trace:

- Performing an LSP ping to an LDP tunnel endpoint
- Performing an LSP trace for an LDP tunnel
- Performing an LSP ping to a segment routing prefix
- Performing an LSP trace to a segment routing prefix
- Performing an LSP ping to an uncolored SR-MPLS TE policy
- Performing an LSP trace to an uncolored SR-MPLS TE policy

4.2.1.1 ECMP considerations for LSP ping and LSP trace

If an LSP trace is initiated without the destination IP address, the sender node does not include multi-path information in the Downstream Mapping TLV of the echo request message (multipath type=0). The responder node replies with a Downstream Mapping TLV for each outgoing interface which is part of the ECMP next hop set for the FEC. The sender node selects the first Downstream Mapping TLV to use for subsequent probes one hop further toward the destination.

If an LSP trace is initiated with the destination IP address, the sender node includes the multipath information in the Downstream Mapping TLV in the echo request message (multipath type=8). The **ecmp-interface-select** and **ecmp-next-hop-select** options allow the LER to exercise a specific ECMP path. If both the options are specified, the **ecmp-interface-select** takes precedence. The **ecmp-interface-select** and **ecmp-next-hop-select** options can be used to direct the echo request message at the sender node to be sent out to a specific outgoing interface which is part of an ECMP path set for the FEC.

4.2.1.2 LSP ping and trace for LDP tunnels

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

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

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

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

4.2.1.2.1 Performing an LSP ping to an LDP tunnel endpoint

Procedure

To check connectivity to an LDP tunnel endpoint, use the **tools oam lsp-ping ldp** command, specifying the IPv4 and IPv6 FEC prefix of the LDP tunnel. To display the results, use the **info from state oam lsp-ping ldp** command, specifying the session ID output from the **lsp-ping**.

Example: Perform an LSP ping to an LDP tunnel endpoint (IPv4)

```
--{ + running }--[  ]--
# tools oam lsp-ping ldp fec 10.20.1.6/32
/oam/lsp-ping/ldp/fec[prefix=10.20.1.6/32]:
    Initiated LSP Ping to prefix 10.20.1.6/32 with session id 49152
```

Example: Display results of the LSP ping (IPv4)

```
--{ + running }--[  ]--
# info from state oam lsp-ping ldp fec 10.20.1.6/32 session-id 49152
    oam {
        lsp-ping {
            ldp {
                fec 10.20.1.6/32 {
                    session-id 49152 {
                        test-active false
                        statistics {
                            round-trip-time {
                                minimum 4292
                                maximum 4292
                                average 4292
                                standard-deviation 0
                            }
                        }
                        path-destination {
                            ip-address 127.0.0.1
                        }
                        sequence 1 {
                            probe-size 48
                            request-sent true
                            out-interface ethernet-1/33.1
                            reply {
                                received true
                                reply-sender 10.20.1.6
                                udp-data-length 40
                                mpls-ttl 255
                                round-trip-time 4292
                                return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                return-subcode 1
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Perform an LSP ping to an LDP tunnel endpoint (IPv6)

```
--{ + running }--[  ]--
# tools oam lsp-ping ldp fec fc00::a14:106/128
/oam/lsp-ping/ldp/fec[prefix=fc00::a14:106/128]:
    Initiated LSP Ping to prefix fc00::a14:106/128 with session id 49169
```

Example: Display results of the LSP ping (IPv6)

```
--{ + running }--[  ]--
# info from state oam lsp-ping ldp fec fc00::a14:106/128 session-id 49169
    oam {
        lsp-ping {
            ldp {
                fec fc00::a14:106/128 {
                    session-id 49169 {
                        test-active false
                        statistics {
                            round-trip-time {
                                minimum 47539
                                maximum 47539
                                average 47539
                                standard-deviation 0
                            }
                        }
                        path-destination {
                            ip-address ::ffff:127.0.0.0
                        }
                        sequence 1 {
                            probe-size 60
                            request-sent true
                            out-interface ethernet-1/31.1
                            reply {
                                received true
                                reply-sender fc00::a14:106
                                udp-data-length 40
                                mpls-ttl 255
                                round-trip-time 47539
                                return-code replying-router-is-egress-for-fec-at-stack-
depth-n
                                return-subcode 1
                            }
                        }
                    }
                }
            }
        }
    }
```

4.2.1.1.2.2 Performing an LSP trace for an LDP tunnel

Procedure

To trace the path to any midpoint or endpoint of an LDP tunnel, use the **tools oam lsp-trace** command, specifying the IPv4 and IPv6 FEC prefix of the LDP tunnel. To display the results, use the **info from state oam lsp-trace ldp** command, specifying the session ID output from the **lsp-trace**.

Example: Perform an LSP trace to an LDP tunnel endpoint (IPv4)

```
--{ + running }--[  ]--
# tools oam lsp-trace ldp fec 10.20.1.6/32
/oam/lsp-trace/ldp/fec[prefix=10.20.1.6/32]:
    Initiated LSP Trace to prefix 10.20.1.6/32 with session id 49153
```

Example: Display results of the LSP trace (IPv4)

```
--{ + running }--[  ]--
# info from state oam lsp-trace ldp fec 10.20.1.6/32 session-id 49153
    oam {
        lsp-trace {
            ldp {
                fec 10.20.1.6/32 {
                    session-id 49153 {
                        test-active false
                        path-destination {
                            ip-address 127.0.0.1
                        }
                        hop 1 {
                            probe 1 {
                                probe-size 76
                                probes-sent 1
                                reply {
                                    received true
                                    reply-sender 10.20.1.2
                                    udp-data-length 60
                                    mpls-ttl 1
                                    round-trip-time 4824
                                    return-code label-switched-at-stack-depth-n
                                    return-subcode 1
                                }
                                downstream-detailed-mapping 1 {
                                    mtu 1500
                                    address-type ipv4-numbered
                                    downstream-router-address 10.10.4.4
                                    downstream-interface-address 10.10.4.4
                                    mpls-label 1 {
                                        label 2002
                                        protocol ldp
                                    }
                                }
                            }
                        }
                        hop 2 {
                            probe 1 {
                                probe-size 76
                                probes-sent 1
                                reply {
                                    received true
                                    reply-sender 10.20.1.4
                                    udp-data-length 60
                                    mpls-ttl 2
                                    round-trip-time 4693
                                    return-code label-switched-at-stack-depth-n
                                    return-subcode 1
                                }
                                downstream-detailed-mapping 1 {
                                    mtu 1500
                                    address-type ipv4-numbered
                                    downstream-router-address 10.10.9.6
                                    downstream-interface-address 10.10.9.6
```

```
                                    mpls-label 1 {
                                        label 2000
                                        protocol ldp
                                    }
                                }
                            }
                            hop 3 {
                                probe 1 {
                                    probe-size 76
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender 10.20.1.6
                                        udp-data-length 32
                                        mpls-ttl 3
                                        round-trip-time 4597
                                        return-code replying-router-is-egress-for-fec-at-
stack-depth-n
                                        return-subcode 1
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Perform an LSP trace to an LDP tunnel endpoint (IPv6)

```
--{ + running }--[  ]--
# tools oam lsp-trace ldp fec fc00::a14:106/128
/oam/lsp-trace/ldp/fec[prefix=fc00::a14:106/128]:
    Initiated LSP Trace to prefix fc00::a14:106/128 with session id 49168
```

Example: Display results of the LSP trace (IPv6)

```
--{ + running }--[  ]--
# info from state oam lsp-trace ldp fec fc00::a14:106/128 session-id 49168
    oam {
        lsp-trace {
            ldp {
                fec fc00::a14:106/128 {
                    session-id 49168 {
                        test-active false
                        path-destination {
                            ip-address ::ffff:127.0.0.0
                        }
                        hop 1 {
                            probe 1 {
                                probe-size 112
                                probes-sent 1
                                reply {
                                    received true
                                    reply-sender fc00::a14:102
                                    udp-data-length 84
                                    mpls-ttl 1
                                    round-trip-time 41527
                                    return-code label-switched-at-stack-depth-n
                                    return-subcode 1
                                }
```

```
                                downstream-detailed-mapping 1 {
                                    mtu 1500
                                    address-type ipv6-numbered
                                    downstream-router-address fe80::201:4ff:feff:1e
                                    downstream-interface-address fe80::201:4ff:feff:1e
                                    mpls-label 1 {
                                        label 2008
                                        protocol ldp
                                    }
                                }
                            }
                        }
                        hop 2 {
                            probe 1 {
                                probe-size 112
                                probes-sent 1
                                reply {
                                    received true
                                    reply-sender fc00::a14:104
                                    udp-data-length 84
                                    mpls-ttl 2
                                    round-trip-time 76569
                                    return-code label-switched-at-stack-depth-n
                                    return-subcode 1
                                }
                                downstream-detailed-mapping 1 {
                                    mtu 1500
                                    address-type ipv6-numbered
                                    downstream-router-address fe80::201:6ff:feff:3
                                    downstream-interface-address fe80::201:6ff:feff:3
                                    mpls-label 1 {
                                        label 2001
                                        protocol ldp
                                    }
                                }
                            }
                        }
                        hop 3 {
                            probe 1 {
                                probe-size 112
                                probes-sent 1
                                reply {
                                    received true
                                    reply-sender fc00::a14:106
                                    udp-data-length 32
                                    mpls-ttl 3
                                    round-trip-time 41739
                                    return-code replying-router-is-egress-for-fec-at-
stack-depth-n
                                    return-subcode 1
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

4.2.1.3 LSP ping and trace for segment routing tunnels

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

- **tools oam lsp-ping sr-isis prefix-sid** *<prefix>*
- **tools oam lsp-trace sr-isis prefix-sid** *<prefix>*

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

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

In the case of ECMP, even when the destination IP is configured, the SR Linux node may not exercise all NHLFEs.

4.2.1.3.1 Performing an LSP ping to a segment routing prefix

Procedure

To check connectivity to a segment routing prefix, use the **tools oam lsp-ping sr-isis** command. To display the results, use the **info from state oam lsp-ping sr-isis** command, specifying the session ID output from the **lsp-ping**.

Example: Perform an LSP ping to a destination segment routing prefix

```
# tools oam lsp-ping sr-isis prefix-sid 10.20.1.6/32
/oam/lsp-ping/sr-isis/prefix-sid[prefix=10.20.1.6/32]:
    Initiated LSP Ping to prefix 10.20.1.6/32 with session id 49152
```

Example: Display results of the LSP ping

```
--{ + running }--[  ]--
# info from state oam lsp-ping sr-isis prefix-sid 10.20.1.6/32 session-id 49152
    oam {
        lsp-ping {
            sr-isis {
                prefix-sid 10.20.1.6/32 {
                    session-id 49152 {
                        test-active false
                        statistics {
                            round-trip-time {
                                minimum 4292
                                maximum 4292
                                average 4292
                                standard-deviation 0
                            }
                        }
                        path-destination {
                            ip-address 127.0.0.1
                        }
                        sequence 1 {
                            probe-size 48
                            request-sent true
                            out-interface ethernet-1/33.1
                            reply {
                                received true
                                reply-sender 10.20.1.6
```

```
                                udp-data-length 40
                                mpls-ttl 255
                                round-trip-time 4292
                                return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                return-subcode 1
                            }
                        }
                    }
                }
            }
        }
    }
```

4.2.1.3.2 Performing an LSP trace to a segment routing prefix

Procedure

To trace the path to any midpoint or endpoint of a segment routing tunnel, use the **tools oam lsp-trace** command. To display the results, use the **info from state oam lsp-trace sr-isis** command, specifying the session ID output from the **lsp-trace**.

Example: Perform an LSP trace to a destination segment routing prefix

```
# tools oam lsp-trace sr-isis prefix-sid 10.20.1.6/32
/oam/lsp-trace/sr-isis/prefix-sid[prefix=10.20.1.6/32]:
    Initiated LSP Trace to prefix 10.20.1.6/32 with session id 49153
```

Example: Display results of the LSP trace

```
--{ + running }--[  ]--
# info from state oam lsp-trace sr-isis prefix-sid 10.20.1.6/32 session-id 49153
    oam {
        lsp-trace {
            sr-isis {
                prefix-sid 10.20.1.6/32 {
                    session-id 49153 {
                        test-active false
                        path-destination {
                            ip-address 127.0.0.1
                        }
                        hop 1 {
                            probe 1 {
                                probe-size 76
                                probes-sent 1
                                reply {
                                    received true
                                    reply-sender 10.20.1.2
                                    udp-data-length 60
                                    mpls-ttl 1
                                    round-trip-time 2768
                                    return-code label-switched-at-stack-depth-n
                                    return-subcode 1
                                }
                                downstream-detailed-mapping 1 {
                                    mtu 1500
                                    address-type ipv4-numbered
                                    downstream-router-address 10.10.4.4
                                    downstream-interface-address 10.10.4.4
                                    mpls-label 1 {
                                        label 27000
```

```
                                    protocol isis
                                }
                            }
                        }
                    }
                    hop 2 {
                        probe 1 {
                            probe-size 76
                            probes-sent 1
                            reply {
                                received true
                                reply-sender 10.20.1.4
                                udp-data-length 60
                                mpls-ttl 2
                                round-trip-time 3414
                                return-code label-switched-at-stack-depth-n
                                return-subcode 1
                            }
                            downstream-detailed-mapping 1 {
                                mtu 1500
                                address-type ipv4-numbered
                                downstream-router-address 10.10.9.6
                                downstream-interface-address 10.10.9.6
                                mpls-label 1 {
                                    label 27000
                                    protocol isis
                                }
                            }
                        }
                    }
                    hop 3 {
                        probe 1 {
                            probe-size 76
                            probes-sent 1
                            reply {
                                received true
                                reply-sender 10.20.1.6
                                udp-data-length 32
                                mpls-ttl 3
                                round-trip-time 4429
                                return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                return-subcode 1
                            }
                        }
                    }
                }
            }
        }
    }
}
```

4.2.1.4 LSP ping and trace for uncolored SR-MPLS TE policy

To check connectivity and trace the path of a segment routing (SR) traffic-engineered (TE) tunnel using an uncolored SR-MPLS TE policy, SR Linux supports the following OAM commands:

- **tools oam lsp-ping te-policy sr-uncolored policy** *<policy name>* **protocol-origin** *<value>*
- **tools oam lsp-trace te-policy sr-uncolored policy** *<policy-name>* **protocol-origin** *protocol-origin <value>*

Supported parameters include **destination-ip**, **source-ip**, **interval**, **segment-list-index**, **timeout**, **ecmp-interface-select mpls-ttl**, **ecmp-next-hop-select**, **send-count**, **traffic-class**, and **probe-size**. The mandatory parameters are, **policy** and **protocol-origin**.

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

4.2.1.4.1 Performing an LSP ping to an uncolored SR-MPLS TE policy

Procedure

To check connectivity to a SR-TE tunnel using an uncolored SR-MPLS TE policy, execute the **tools oam lsp-ping te-policy sr-uncolored policy** command, specifying the uncolored SR-MPLS TE policy name and the protocol origin. To display the results, use the **info from state oam lsp-ping te-policy sr-uncolored policy protocol-origin session-id** command, specifying the session ID output from the **lsp-ping**.

Example: Perform an LSP ping to an uncolored SR-MPLS TE policy

```
--{ + running }--[  ]--
# tools oam lsp-ping te-policy sr-uncolored policy polABCEF protocol-origin local
/:
Initiated LSP Ping for TE-policy polABCEF with session id 49152.
Please check "info from state oam" for result
```

Example: Display results of the LSP ping

```
--{ + running }--[  ]--
# info from state oam lsp-ping te-policy sr-uncolored policy polABCEF protocol-origin local session-id
 49152
    oam {
        lsp-ping {
            te-policy {
                sr-uncolored {
                    policy polABCEF protocol-origin local {
                        session-id 49152 {
                            test-active false
                            statistics {
                                round-trip-time {
                                    minimum 83
                                    maximum 83
                                    average 83
                                    standard-deviation 0
                                }
                            }
                            path-destination {
                                ip-address 127.0.0.1
                            }
                            sequence 1 {
                                probe-size 64
                                request-sent true
                                out-interface ethernet-1/31.1
                                reply {
                                    received true
                                    reply-sender 10.20.1.6
                                    udp-data-length 40
                                    mpls-ttl 255
                                    round-trip-time 83
                                    return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                    return-subcode 1
```

```
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

4.2.1.4.2 Performing an LSP trace to an uncolored SR-MPLS TE policy

Procedure

To trace the path of a SR-TE tunnel using an uncolored SR-MPLS TE policy, execute the **tools oam lsp-trace te-policy sr-uncolored policy** command, specifying the uncolored SR-MPLS TE policy name and the protocol origin. To display the results, use the **info from state oam lsp-trace te-policy sr-uncolored policy protocol-origin session-id** command, specifying the session ID output from the **lsp-trace**.

Example: Perform an LSP trace to an uncolored SR-MPLS TE policy

```
--{ + running }--[  ]--
# tools oam lsp-trace te-policy sr-uncolored policy polABCEF protocol-origin local
/:
Initiated LSP Trace for TE-policy polABCEF with session id 49153.
Please check "info from state oam" for result
```

Example: Display results of the LSP trace

```
--{ + running }--[  ]--
# info from state oam lsp-trace te-policy sr-uncolored policy polABCEF protocol-origin local session-
id 49153
    oam {
        lsp-trace {
            te-policy {
                sr-uncolored {
                    policy polABCEF protocol-origin local {
                        session-id 49153 {
                            test-active false
                            path-destination {
                                ip-address 127.0.0.1
                            }
                            hop 1 {
                                probe 1 {
                                    probe-size 188
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender 10.20.1.2
                                        udp-data-length 32
                                        mpls-ttl 1
                                        round-trip-time 165172
                                        return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                        return-subcode 4
                                    }
                                }
                                probe 2 {
                                    probe-size 156
                                    probes-sent 1
```

```
                                reply {
                                    received true
                                    reply-sender 10.20.1.2
                                    udp-data-length 68
                                    mpls-ttl 1
                                    round-trip-time 54673
                                    return-code label-switched-at-stack-depth-n
                                    return-subcode 3
                                }
                                downstream-detailed-mapping 1 {
                                    mtu 1500
                                    address-type ipv4-numbered
                                    downstream-router-address 10.10.3.3
                                    downstream-interface-address 10.10.3.3
                                    mpls-label 1 {
                                        label IMPLICIT_NULL
                                        protocol isis
                                    }
                                    mpls-label 2 {
                                        label 70019
                                        protocol isis
                                    }
                                    mpls-label 3 {
                                        label 70009
                                        protocol isis
                                    }
                                }
                            }
                        }
                        hop 2 {
                            probe 1 {
                                probe-size 156
                                probes-sent 1
                                reply {
                                    received true
                                    reply-sender 10.20.1.3
                                    udp-data-length 32
                                    mpls-ttl 2
                                    round-trip-time 103751
                                    return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                    return-subcode 3
                                }
                            }
                            probe 2 {
                                probe-size 124
                                probes-sent 1
                                reply {
                                    received true
                                    reply-sender 10.20.1.3
                                    udp-data-length 64
                                    mpls-ttl 2
                                    round-trip-time 8262
                                    return-code label-switched-at-stack-depth-n
                                    return-subcode 2
                                }
                                downstream-detailed-mapping 1 {
                                    mtu 1500
                                    address-type ipv4-numbered
                                    downstream-router-address 10.10.5.5
                                    downstream-interface-address 10.10.5.5
                                    mpls-label 1 {
                                        label IMPLICIT_NULL
                                        protocol isis
                                    }
```

```
                                    mpls-label 2 {
                                        label 70009
                                        protocol isis
                                    }
                                }
                            }
                            hop 3 {
                                probe 1 {
                                    probe-size 124
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender 10.20.1.5
                                        udp-data-length 32
                                        mpls-ttl 3
                                        round-trip-time 57971
                                        return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                        return-subcode 2
                                    }
                                }
                                probe 2 {
                                    probe-size 92
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender 10.20.1.5
                                        udp-data-length 60
                                        mpls-ttl 3
                                        round-trip-time 101694
                                        return-code label-switched-at-stack-depth-n
                                        return-subcode 1
                                    }
                                    downstream-detailed-mapping 1 {
                                        mtu 1500
                                        address-type ipv4-numbered
                                        downstream-router-address 10.10.10.6
                                        downstream-interface-address 10.10.10.6
                                        mpls-label 1 {
                                            label IMPLICIT_NULL
                                            protocol isis
                                        }
                                    }
                                }
                            }
                            hop 4 {
                                probe 1 {
                                    probe-size 92
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender 10.20.1.6
                                        udp-data-length 32
                                        mpls-ttl 4
                                        round-trip-time 14393
                                        return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                        return-subcode 1
                                    }
                                }
                            }
                        }
                    }
                }
            }
```

```
        }
    }
```

4.2.1.5 LSP ping and trace for colored SR-MPLS TE policy

Note: This feature is supported on 7730 SXR, 7250 IXR Gen 2c+, and 7250 IXR Gen 2 platforms.

To check connectivity or trace the path of a traffic engineered (TE) tunnel using a colored SR-MPLS TE policy, SR Linux supports the following OAM commands:

- **tools oam lsp-ping te-policy sr-colored policy** <*color*> **endpoint** <*value*>
- **tools oam lsp-trace te-policy sr-colored policy** <*color*> **endpoint** <*value*> **discriminator** <*value*> **protocol-origin** <*value*> **originator-asn** <*value*> **originator-address** <*value*>

4.2.1.5.1 Performing an LSP ping to a colored SR-MPLS TE policy

Procedure

To check connectivity of a colored SR-MPLS TE policy, execute the **tools oam lsp-ping te-policy sr-colored policy** command, specifying the colored SR-MPLS TE policy color and the endpoint. To display the results, use the **info from state oam lsp-ping te-policy sr-colored policy endpoint session-id** command, specifying the session ID output from the **lsp-ping**.

Example: Perform an LSP ping to a colored SR-MPLS TE policy

This example performs an LSP ping to a colored SR-MPLS TE policy.

```
--{ + running }--[  ]--
# tools oam lsp-ping te-policy sr-colored policy 1 endpoint 10.20.1.6
/:
Initiated LSP Ping for TE-policy (endpoint 10.20.1.6 and color 1) with session id 49154.
Please check "info from state oam" for result
```

Example: Display results of the LSP ping

This example displays the results of the LSP ping.

```
--{ + running }--[  ]--
# info from state oam lsp-ping te-policy sr-colored policy 1 endpoint 10.20.1.6 session-id 49154
    oam {
        lsp-ping {
            te-policy {
                sr-colored {
                    policy 1 endpoint 10.20.1.6 {
                        session-id 49154 {
                            test-active false
                            statistics {
                                round-trip-time {
                                    minimum 50
                                    maximum 50
                                    average 50
                                    standard-deviation 0
                                }
```

```
                            }
                            path-destination {
                                ip-address 127.0.0.1
                            }
                            sequence 1 {
                                probe-size 48
                                request-sent true
                                out-interface ethernet-1/31.1
                                reply {
                                    received true
                                    reply-sender 10.20.1.6
                                    udp-data-length 40
                                    mpls-ttl 255
                                    round-trip-time 50
                                    return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                    return-subcode 1
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

4.2.1.5.2 Performing an LSP trace to a colored SR-MPLS TE policy

Procedure

To trace the path of a colored SR-MPLS TE policy, execute the **tools oam lsp-trace te-policy sr-colored policy** command, specifying the colored SR-MPLS TE policy color, endpoint, discriminator, protocol origin, originator ASN, and originator IP address. To display the results, use the **info from state oam lsp-trace te-policy sr-colored policy protocol-origin session-id** command, specifying the session ID output from the **lsp-trace**.

Example: Perform an LSP trace to a colored SR-MPLS TE policy with an IPv6 endpoint

```
--{ + running }--[  ]--
# tools oam lsp-trace te-policy sr-colored policy 1 endpoint fc00::a14:106
/:
Initiated LSP Trace for TE-policy with endpoint fc00::a14:106 and color 1 with session id 49153.
Please check "info from state oam" for result
```

Example: Display results of the LSP trace to a colored SR-MPLS TE policy with an IPv6 endpoint

```
--{ + running }--[  ]--
# info from state oam lsp-trace te-policy sr-colored policy 1 endpoint fc00::a14:106 session-id 49153
    oam {
        lsp-trace {
            te-policy {
                sr-colored {
                    policy 1 endpoint fc00::a14:106 {
                        session-id 49153 {
                            test-active false
                            path-destination {
                                ip-address ::ffff:127.0.0.0
                            }
```

```
                            hop 1 {
                                probe 1 {
                                    probe-size 112
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender fc00::a14:102
                                        udp-data-length 84
                                        mpls-ttl 1
                                        round-trip-time 48804
                                        return-code label-switched-at-stack-depth-n
                                        return-subcode 1
                                    }
                                    downstream-detailed-mapping 1 {
                                        mtu 1500
                                        address-type ipv6-numbered
                                        downstream-router-address fe80::201:4ff:feff:1e
                                        downstream-interface-address fe80::201:4ff:feff:1e
                                        mpls-label 1 {
                                            label 16500
                                            protocol isis
                                        }
                                    }
                                }
                            }
                            hop 2 {
                                probe 1 {
                                    probe-size 112
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender fc00::a14:104
                                        udp-data-length 84
                                        mpls-ttl 2
                                        round-trip-time 30653
                                        return-code label-switched-at-stack-depth-n
                                        return-subcode 1
                                    }
                                    downstream-detailed-mapping 1 {
                                        mtu 1500
                                        address-type ipv6-numbered
                                        downstream-router-address fe80::201:6ff:feff:3
                                        downstream-interface-address fe80::201:6ff:feff:3
                                        mpls-label 1 {
                                            label 16500
                                            protocol isis
                                        }
                                    }
                                }
                            }
                            hop 3 {
                                probe 1 {
                                    probe-size 112
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender fc00::a14:106
                                        udp-data-length 32
                                        mpls-ttl 3
                                        round-trip-time 12664
                                        return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                        return-subcode 1
                                    }
                                }
```

```
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Perform an LSP trace to a colored SR-MPLS TE policy with an IPv4 endpoint

This example performs an LSP trace to a colored SR-MPLS TE policy.

```
--{ + running }--[  ]--
# tools oam lsp-trace te-policy sr-colored policy 1 endpoint 10.20.1.6 discriminator 4116 protocol-
origin local originator-asn 0 originator-address 0.0.0.0
/:
Initiated LSP Trace for TE-policy witn endpoint 10.20.1.6 color 1 protocol-origin local originator-asn
 0 originator-address 0.0.0.0 discriminator 4116 with session id 49152.
Please check "info from state oam" for result
```

Example: Display results of the LSP trace to a colored SR-MPLS TE policy with an IPv4 endpoint

This example displays the results of the LSP trace.

```
--{ + running }--[  ]--
A:Dut-A# info from state oam lsp-trace te-policy sr-colored policy 1 endpoint 10.20.1.6 session-id
 49152
    oam {
        lsp-trace {
            te-policy {
                sr-colored {
                    policy 1 endpoint 10.20.1.6 {
                        session-id 49152 {
                            test-active false
                            path-destination {
                                ip-address 127.0.0.1
                            }
                            hop 1 {
                                probe 1 {
                                    probe-size 76
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender 10.20.1.2
                                        udp-data-length 60
                                        mpls-ttl 1
                                        round-trip-time 46110
                                        return-code label-switched-at-stack-depth-n
                                        return-subcode 1
                                    }
                                    downstream-detailed-mapping 1 {
                                        mtu 1500
                                        address-type ipv4-numbered
                                        downstream-router-address 10.10.4.4
                                        downstream-interface-address 10.10.4.4
                                        mpls-label 1 {
                                            label 16000
                                            protocol isis
                                        }
                                    }
                                }
```

```
                            }
                            hop 2 {
                                probe 1 {
                                    probe-size 76
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender 10.20.1.4
                                        udp-data-length 60
                                        mpls-ttl 2
                                        round-trip-time 9671
                                        return-code label-switched-at-stack-depth-n
                                        return-subcode 1
                                    }
                                    downstream-detailed-mapping 1 {
                                        mtu 1500
                                        address-type ipv4-numbered
                                        downstream-router-address 10.10.9.6
                                        downstream-interface-address 10.10.9.6
                                        mpls-label 1 {
                                            label 16000
                                            protocol isis
                                        }
                                    }
                                }
                            }
                            hop 3 {
                                probe 1 {
                                    probe-size 76
                                    probes-sent 1
                                    reply {
                                        received true
                                        reply-sender 10.20.1.6
                                        udp-data-length 32
                                        mpls-ttl 3
                                        round-trip-time 11661
                                        return-code replying-router-is-egress-for-fec-at-stack-depth-n
                                        return-subcode 1
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}
```

4.3 Ethernet OAM tools and protocols

This section provides information about the Ethernet OAM tools and protocols.

4.3.1 Ethernet connectivity fault management

Note: This feature is supported on 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, 7250 IXR-X3b, 7730 SXR and 7250 IXR Gen 3 platforms.

Ethernet connectivity fault management (ETH-CFM) is a set of Ethernet-layer OAM protocols that provide capabilities to detect, verify, isolate, and report Ethernet connectivity faults. The connectivity and fault management functions are co-defined by IEEE802.1ag and ITU-T Y.1731.

4.3.1.1 ETH-CFM components

Naming conventions

The IEEE and the ITU-T use their own nomenclature to describe the administrative contexts and functions. This introduces a level of complexity to configuration, discussion, and vendor's naming conventions. Table 7: Naming conventions lists the IEEE and the ITU-T nomenclature of the ETH-CFM components.

Table 7: Naming conventions

IEEE 802.1ag naming	ITU-T Y.1731 naming	Function
Maintenance Domain (MD)	Maintenance Entity (ME)	Administrative scope and reach
Maintenance Domain Level	MEG Level	Numerical identifier of the domain
Maintenance Association (MA)	Maintenance Entity Group (MEG)	Grouping of service endpoints
Maintenance Association Endpoint (MEP)	MEG Endpoint (MEP)	Terminating and origination endpoints

Maintenance domain

Maintenance domain (MD) or Maintenance entity (ME) is the administrative container that defines the scope, reach, and boundary for testing and faults. It is the area of ownership and management responsibility.

SR Linux supports the following domain name formats:

- none
- DNS name string
- MAC address
- string

See Configuring a maintenance domain for more information.

Maintenance domain level

Maintenance domain level (MD level) or Maintenance entity group level (MEG level) is the numerical value (0-7) representing the width of the domain. The wider the domain or higher the numerical value, the farther the ETH-CFM packets can travel. The level establishes the processing boundary for the packets. ETH-

CFM packets with higher numerical level values flow through MEPs and MIPs configured with equal or lower level values.

See Configuring a maintenance domain for more information.

Maintenance association

Maintenance association (MA) or Maintenance entity group (MEG) is the construct where the different management entities are contained. Each MA is uniquely identified by its MA-ID. The MA-ID comprises the MD level, MA name, and associated format. The MA short name formats (0 to 255) are divided between the IEEE (0 to 31, 64 to 255) and the ITU-T (32 to 63), with five currently defined (1 to 4, 32). Even though the different standards bodies do not have specific support for the other's formats, the domain and association formats can be mixed. The interpretation of the information in the PDU is based on the domain format.

The following formats are supported:

- vid
- string
- integer
- VPN ID
- icc-format

See Configuring a maintenance association for more information.

Maintenance association endpoints

Maintenance Association Endpoints (MEP) or MEG Endpoints (MEP) are active ETH-CFM entities that initiate, process, and terminate ETH-CFM functions while following the domain nesting rules. MEPs are configured on the Ethernet interfaces, Ethernet subinterfaces, and LAG subinterfaces. LAG subinterface support is only available on 7730 SXR platforms.

A MEP is the unique identifier within the association. Each MEP is uniquely identified by the MA-ID and MEP-ID tuple. This management entity is responsible for initiating, processing, and terminating ETH-CFM packets. MEPs form boundaries that prevent the ETH-CFM packets from flowing beyond their specific scope of responsibility. MEPs within the same MA and at the same level (MA-ID) represent points within a single network instance. The number of MEPs per MA is limited to 64. Each local MEP maintains its own remote MEP database. If there are two Up MEPs in mac-vrf-1 and a remote mac-vrf has a single MEP, the MEP count for the mac-vrf-1 equals six. Each of the local MEPs in the mac-vrf-1 has three entries: itself, the other local Up MEP, and the remote mac-vrf MEP. Exceeding this value generates an error indicating that the remote MEP or local MEP cannot be added because the maximum limit has reached.

A MEP has an **up** or **down** direction. It indicates the directions that the packets are generated. The up MEP generates packets toward the switch fabric. The down MEP generates packets toward the line away from the fabric.

A MEP has an active and passive side. Packets that enter the active side of the MEP are compared to the existing MD level and processed accordingly where equal or lower levels are terminated and processed. Higher levels transparently pass through. Packets that enter the passive side of the MEP are exposed to the same processing rules as the active side. However, the passive side of the MEP silently discards those levels that are equal to or lower without processing.

See Configuring a maintenance association endpoint for more information.

4.3.1.2 Automatically discover remote MEPs

Remote MEPs can be statically configured as association MEPs using the **oam ethcfm domain association association-meps** command. As soon as the static remote MEPs are added, the process for timing out the MEP commences.

The MEPs automatically add remote MEPs to the remote MEP database when you enable the **remote-mep-auto-discovery** command at the maintenance association context. The ETH-CCM packets trigger an addition of the peer MEP ID to the remote MEP database. The remote MEPs added through auto-discovery are uniquely identified. These auto-discovered MEPs can be configured with an **aging-timer** to remove them from the remote MEP database following a timeout condition that lasts for the duration of the aging timer. These auto-discovered MEPs appear in the MA as association MEPs. These remote MEPs are treated the same way as the statically configured MEPs after they are discovered. When auto-discovery is enabled for remote MEPs, the unexpected connectivity defects are not identified and the **defXcon** defects are not raised.

See Configuring remote MEP auto-discovery for more information.

4.3.1.3 Remote MEP ID to MAC address resolution

The Ethernet continuity check messages (ETH-CCMs) contain the pertinent information about the peer. The source Ethernet address in a received ETH-CCM packet is used to generate an entry in the remote MEP-ID to MAC address mapping table for the receiving local MEPs within the same MA. ETH-CFM test protocols, such as ETH-LBM (loopback) and ETH-LTM (linktrace), require a target. This remote MEP-ID resolution table allows the test to target the remote MEP-ID instead of the Layer-2 Ethernet MAC address. A lookup is performed for the local MEP to resolve the remote MEP ID to a Layer-2 MAC address. If the lookup is successful, the test uses the Layer 2 MAC address as the destination address in the Ethernet header. If the lookup is unsuccessful, the ETH-CFM test fails to initiate.

You can clear the automatically learned MAC address of remote MEPs from the Layer 2 address resolution mapping table at multiple levels of the hierarchy. See Clearing the learned remote MEP MAC address for more information.

4.3.1.4 MAC address assignment to MPs

The allocation of MAC addresses to maintenance points (MPs) varies depending on platform capabilities and configuration modes. The default MP MAC address assignment is platform-specific. A hierarchical preference model is available to select the appropriate MAC address configuration for each MP, based on the configured allocation mode.

Note: MAC address duplication must be avoided when considering learning bridge connections. Configuring the same Ethernet MAC address on two different MEPs in a learning bridge causes MAC moves and connectivity issues.

MAC address allocation modes in 7250 IXR platforms

The 7250 IXR platform restricts allocation of MP MAC addressing. The first five-byte prefix (40-bits) of the MEP MAC address must be the same on individual Ethernet interfaces, including all subinterfaces on that Ethernet interface. A maximum of two prefixes may exist on any single linecard. When the Ethernet interfaces are LAG member interfaces, all ports in that LAG must have the same five byte prefix. Violations

of these rules causes the MEPs that are in violation of the rules to fail the processing of any unicast ETH-CFM message.

The following are the MAC address allocation modes in 7250 IXR platforms that determine the MAC address allocation strategy for the node:

- **oam ethcfm mac-allocation mode port** {platform default}
- **oam ethcfm mac-allocation mode mac-pool**

  An MP can use one of the following two MAC address pools per line card:

  – **custom-mac-pool**

  – system-generated MAC address pool

Note: To ensure flexibility and support of future functions, Nokia recommends that you deploy MPs using the mac-allocation mode, **mac-pool** for 7250 IXR platforms. This avoids migration techniques in the future.

Note: MPs are not supported on LAG subinterface on the 7250 IXR platforms. As a pre-deployment consideration, when these types of subinterfaces are supported, they would have a node-global scope. This would require the use of MAC address pools for any MEP on a LAG subinterface. If the existing deployment uses the default MAC allocation mode **port**, this would need to be changed to **mac-pool** before deploying MEPs on subinterfaces over LAG.

MAC address allocation modes in 7730 SXR platforms

The 7730 SXR platforms do not restrict the MAC addressing for MPs. The following are the MAC address allocation modes in 7730 SXR platforms that determine the MAC address allocation strategy for the entire network:

- **oam ethcfm mac-allocation mode port**
- **oam ethcfm mac-allocation mode mac-pool**
- **oam ethcfm mac-allocation mode any** {platform default}

  MP uses one of the following two MAC address pools:

  – **custom-mac-pool**

  – system-generated MAC address pool

- **Note:** Because the 7730 SXR platforms do not restrict the MP MAC addressing, Nokia recommends that you do not change the default MAC allocation mode, **any** for 7730 SXR platforms. This provides maximum flexibility.

MAC address allocation mode - port

When you configure the MAC address allocation mode as **port**, the MP uses the port or LAG MAC address advertised by the hardware. The MAC address is derived from that pool hosting the MEP. Any **mac-pool** or **custom-address** configurations under the MP are ignored when the MAC allocation mode is set to port.

MAC address allocation mode - mac-pool

For MPs that require a custom MAC address, you can configure a user-defined pool with a starting MAC address and a count of the number of contiguous MAC addresses in the custom MAC address pool. The last byte of the starting MAC address is incremented to create a list of unique MAC addresses for the pool. This custom pool defines the available range of MAC addresses.

The ETH-CFM application on an 7250 IXR platform ensures that the combination of **starting-mac** and the **count** in the custom pool does not cause any change to the first five bytes of the MAC address. If the **starting-mac** and **count** exceed the last byte FF boundary, an error message is displayed indicating that this is not a valid configuration.

For 7730 SXR platforms, if the **starting-mac** and **count** exceed the last byte FF boundary, the MAC addresses increase the first five-byte prefix rolling over the next available incremental hexadecimal value. The next available hexadecimal value in the first five byte prefix may not be in the last byte of that prefix. Changes to the first byte of a MAC address could have implications for the global/local, or unicast/multicast bits. If the combination of **starting-mac** and **count** would cause a change to the first byte of the MAC address, the configuration is rejected.

The system-generated MAC address pool is reserved for ETH-CFM. This pool has a fixed count of 64 MAC addresses with the same first five-byte prefix.

The hierarchical preference is as follows: **custom-mac-pool**, system-generated MAC address pool. If neither of the pools are configured, the ETH-CFM application does not allow the **mac-pool** mode to be configured. An error message, `ETHCFM mac-allocation mode setting inconsistent with MP configuration` is displayed.

MAC address allocation mode - any

7730 SXR platforms offer flexibility with no restrictions on MAC address assignment, supporting a configurable custom MAC address, **mac-pool**, or **any** allocation mode that allows you to mix different allocation schemes. The hierarchical preference is as follows: custom MAC address, custom MAC pool address, system MAC pool address, port hardware address.

4.3.1.5 ETH-CFM statistics

The 7730 SXR platforms support system-level and MEP-level statistics with a breakdown for each OpCode. The system maintains an aggregate view of CFM statistics on the node. Every MEP on the system maintains statistics for each OpCode.

See Displaying ETH-CFM statistics for more information.

These statistics are cleared using **tools** commands.

See Clearing ETH-CFM system statistics and Clearing ETH-CFM statistics for each MEP for more information.

4.3.2 Ethernet continuity check message

Note: This feature is supported on 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, 7250 IXR-X3b, 7250 IXR Gen 3, and 7730 SXR platforms.

An Ethernet continuity check message (ETH-CCM) is a multicast frame that is used for fault detection, notification, and recovery. A CCM is generated by a MEP at a configurable periodic interval and multicast to all other MEPs in the same MA. This is not an echo request/echo response model but a model that multicasts to all endpoints along the datapath. To identify faults, the receiving MEP maintains an internal list of all of the remote MEPs from which it should receive the CCMs. This list is based on the remote MEP ID configuration within the MA. When the local MEP does not receive a CCM from one of the configured remote MEPs within a preconfigured period, it raises an alarm. Several other defect conditions can be detected and the detection of the these defect conditions is specific to the SR Linux platform.

See Configuring ETH-CCM for more information.

4.3.2.1 ETH-CCM hold time

ETH-CCM hold time prevents a MEP from timing out a peer (**defRemoteCCM**) until an additional time (**ccm-hold-time**) has elapsed.

The IEEE 802.1ag standard and ITU-T Y.1731 recommendations specify a non-configurable timeout of 3.5 times the CCM interval to determine if a peer has timed out. When CCM is enabled, the **ccm-hold-time delay-timeout** option provides additional time before declaring a peer lost because of timeout. If a CCM arrives before the delay-timeout reaches zero, the peer does not timeout.

See Configuring ETH-CCM for more information about configuring the **ccm-hold-time**.

4.3.2.2 Down MEP CCM local fault action

When a down MEP CCM experiences a connectivity fault equal to or greater than the local MEP's lowest fault priority defect, the local fault action will operationally affect the interface or subinterface on which the MEP is configured.

You can enable or disable this action by configuring the **ccm-local-fault** and specifying the local fault action to be either **permit** or **deny** respectively. The behaviour is as follows:

- When the **ccm-local-fault** is configured as **permit**, the CCM defects affect the operational state of the interface or subinterface on which the MEP is configured.
- When the **ccm-local-fault** is configured as **deny**, a CCM defect, regardless of its priority does not negatively affect the operational state of the interface or subinterface on which the MEP is configured.
- When transitioning the **ccm-local-fault** configuration from **permit** to **deny**, any negative operational effects on the interface or subinterface are removed.
- When transitioning the **ccm-local-fault** configuration from **deny** to **permit**, any local MEP fault where the transition is configured is analyzed and necessary action is taken on the interface or subinterface on which the MEP is configured

See Configuring ETH-CCM for more information about configuring the local fault action.

4.3.2.3 Ethernet remote defect indication

The Ethernet Remote Defect Indication function (ETH-RDI) is used by a MEP to communicate to its peer MEPs that a defect condition has been encountered. Defect conditions such as signal failure may result in the transmission of frames with ETH-RDI information. ETH-RDI is used only when ETH-CCM transmission is enabled. ETH-RDI is a bit in the CCM PDU.

ETH-RDI has the single-ended fault management application. The receiving MEP detects an RDI defect condition, which gets correlated with other defect conditions for this MEP. The absence of received ETH-RDI on all MEPs in the association indicates the absence of defects in the entire MEG.

A MEP that is in a defect condition transmits frames with ETH-RDI information. A MEP, upon receiving frames with ETH-RDI information, determines that its peer MEP has encountered a defect condition.

4.3.3 Ethernet linktrace

Ethernet linktrace messages (ETH-LTMs) are multicast frames transmitted by a MEP to trace the path (hop-by-hop) to a peer MEP in the same MA. The peer MEP responds with an Ethernet linktrace reply message (ETH-LTR) after successfully inspecting the linktrace message. If the received linktrace message that has a TTL greater than one, a lookup is performed for the target unicast MAC in the LTM PDU. The message is then processed (responded to and forwarded) along the path if the target MAC address is found in the FDB.

Ethernet linktrace tests are performed using the **tools oam ethcfm domain association mep on-demand linktrace** command. See Performing an Ethernet CFM linktrace test for more information. A MEP can only have a single active LTM test executing at a specific time. A MEP stores one test result, and the subsequent test results are overwritten.

Note: SR Linux does not support MIPs.

4.3.4 Ethernet loopback

Ethernet loopback messages (ETH-LBMs) are unicast or multicast frames transmitted by a MEP to verify connectivity to a particular maintenance point, reporting destination reachability. A MEP responds to ETH-LBMs with loopback reply (ETH-LBR) messages.

Ethernet loopback tests are performed using the **tools oam ethcfm domain association mep on-demand loopback** command. See Performing an Ethernet CFM loopback test for more information. A MEP can only have a single active LBM test executing at a specific time. A MEP stores one test result, and the subsequent test results are overwritten.

4.3.5 Configuring ETH-CFM tools and protocols

This section provides information about how to configure ETH-CFM tools and examples of common configurations.

4.3.5.1 Configuring a maintenance domain

Procedure

To configure a maintenance domain (MD), you specify the domain ID, domain name format, domain name, and the level in the **ethcfm** context. The supported domain name formats include, **none**, **dns-like**, **mac-address**, and **string**. The domain level is used to indicate the nesting relationship between the current domain and other domains. The domain level value ranges from 0 to 7.

Example: Configuring an MD

The following example configures an MD.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm
    oam {
        ethcfm {
```

```
            domain MD1 {
                domain-format string
                level 1
                md-name {
                    name domain1
                }
            }
        }
    }
```

4.3.5.2 Configuring a maintenance association

Procedure

To configure a maintenance association (MA), you specify the MA ID, MA name format and the name. The supported MA name formats include, **vlan-id**, **string**, **integer**, **vpn-id**, and **icc-based**.

Example: Configuring an MA

The following example configures an MA.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm domain MD1
    oam {
        ethcfm {
            domain MD1 {
                domain-format string
                level 1
                md-name {
                    name domain1
                }
                association MA1 {
                    association-format string
                    ma-name {
                        name association1
                    }
                }
            }
        }
    }
```

4.3.5.3 Configuring a maintenance association endpoint

Procedure

Perform the following steps to configure a maintenance association endpoint (MEP):

1. Configure an association MEP by specifying the **network-instance** and the MEP ID of the **association-mep** in the **association** context. If the MEP is non-subinterface based, hosted on the interface, the network-instance is not required. See Configuring an association MEP for an example of an association MEP configuration.
2. Enable the MEP, specify the interface-ref, and direction in the **mep** context.

   The MEP configuration scenarios are described in the following table.

Table 8: MEP configuration scenarios

Platforms	MAC-VRF L2 subinterface	VPWS L2 subinterface	IP-VRF service and default network instance L3 subinterface	Ethernet Interface
7730 SXR	Down MEP and up MEP on L2 subinterfaces including L2 subinterfaces on LAG.	Down MEP and up MEP on L2 subinterfaces including L2 subinterfaces on LAG.	Down MEP on L3 subinterfaces including L3 subinterface on LAG.	Down MEP
7250 IXR Gen 3	Not supported	Not supported	Down MEP on L3 subinterfaces but NOT L3 subinterface on LAG.	Down MEP
7250 IXR Gen 2c+ 7250 IXR-X1b 7250 IXR-X3b	Down MEP and up MEP on L2 subinterfaces but NOT L2 subinterfaces on LAG.	Down MEP and up MEP on L2 subinterfaces but NOT L2 subinterfaces on LAG.	Down MEP on L3 subinterfaces but NOT L3 subinterface on LAG.	Down MEP

See Configuring a local down MEP for an example of a local down MEP configuration.

See Configuring a local up MEP for an example of a local up MEP configuration.

Example: Configuring an association MEP

The following example configures an association MEP. An association MEP is a list of MEPs in the association. The MEP configuration is required to transition one of the association MEPs to a local MEP.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm domain MD1 association MA1
    oam {
        ethcfm {
            domain MD1 {
                association MA1 {
                    association-format string
                    ma-name {
                        name association1
                    }
                    network-instance {
                        name oam-test
                    }
                    association-meps 1 {
                    }
                    association-meps 2 {
                    }
                }
```

```
            }
        }
    }
```

Example: Configuring a local down MEP

The following example configures a local down MEP.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm domain MD1 association MA1
    oam {
        ethcfm {
            domain MD1 {
                association MA1 {
                    association-format string
                    ma-name {
                        name association1
                    }
                    network-instance {
                        name oam-test
                    }
                    association-meps 1 {
                    }
                    association-meps 2 {
                    }
                    mep 1 {
                        admin-state enable
                        direction down
                        interface-ref {
                            interface ethernet-1/10
                            subinterface 3
                        }
                    }
                }
            }
        }
    }
```

Example: Configuring a local up MEP

The following example configures a local up MEP.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm domain MD3 association MA3
    oam {
        ethcfm {
            domain MD3 {
                association MA3 {
                    association-format string
                    ma-name {
                        name MA3
                    }
                    network-instance {
                        name oam-test
                    }
                    association-meps 1 {
                    }
                    mep 1 {
                        direction up
                        interface-ref {
                            interface ethernet-1/20
                            subinterface 2
                        }
```

```
                    }
                }
            }
        }
    }
```

4.3.5.4 Configuring remote MEP auto-discovery

Procedure

There are two methods to add remote MEPs to the remote MEP database:

- Manually configure a MEP. See Configuring a maintenance association endpoint for more information about configuring a MEP.
- Configure remote MEP auto-discovery

To automatically add the remote MEP IDs contained in the received CCM to the remote MEP database, you enable the admin-state of the **remote-mep-auto-discovery** command and optionally specify the **aging-timer** value for a specific **association**.

Example: Configuring remote MEP auto-discovery

The following example configures remote MEP auto discovery.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm domain MD2 association MD2
    oam {
        ethcfm {
            domain MD2 {
                association MD2 {
                    association-format string
                    ma-name {
                        name association2
                    }
                    remote-mep-auto-discovery {
                        admin-state enable
                        aging-timer 86400
                    }
                }
            }
        }
    }
```

4.3.5.5 Configuring ETH-CCM

Procedure

Perform the following steps to enable ETH-CCM and configure the ETH-CCM parameters:

1. Configure a maintenance domain. See Configuring a maintenance domain for more information.
2. Configure a maintenance association. See Configuring a maintenance association for more information.
3. Configure maintenance end points. See Configuring a maintenance association endpoint for more information.

4. To transmit CCM, enable the admin state of the **ccm-transmit** command under the **continuity-check** context.
5. To configure the lowest priority defect for the local MEP, configure the **oam ethcfm domain association mep continuity-check lowest-priority-defect** command.
6. To enable and disable local fault action, configure the **oam ethcfm domain association mep continuity-check ccm-local-fault** command.
7. To generate CCM at periodic intervals, specify the **ccm-interval** time in the **association** context.
8. To configure CCM hold time, specify the time in the **ccm-hold-time** parameter **delay-timeout** option under the **association** context.
9. To configure the sender ID TLV information to be included in the ETH-CCM and ETH-LTM packets, you specify how the Sender ID TLV information is included using the **sender-id-permission-type** command (7730 SXR only).

   Before configuring the **sender-id-permission-type** as **chassis**, ensure that you have performed the global sender ID configuration as follows:

   - Configure the sender ID using the **oam ethcfm sender-id** command.
   - If you configure the **oam ethcfm sender-id chassis-type** as **local**, then specify the local name in the **chassis-local-name** option. See Configuring chassis type as local for sender ID TLV for an example configuration.
   - If you configure the **oam ethcfm sender-id chassis-type** as **system**, then do not configure the **chassis-local-name** option. See Configuring chassis type as system for sender ID TLV for an example configuration.
10. To automatically add remote MEP-IDs in the received CCM, enable the admin-state of the **remote-mep-auto-discovery** command and optionally specify the **aging-timer** value for a specific **association**.

Example: Configuring ETH-CCM (7250 IXR devices)

The following example enables the Ethernet continuity check and configures the ETH-CCM parameters.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm domain MD1 association MA1
    oam {
        ethcfm {
            domain MD1 {
                association MA1 {
                    association-format string
                    ccm-interval 10ms
                    ma-name {
                        name association1
                    }
                    network-instance {
                        name test
                    }
                    ccm-hold-time {
                        delay-timeout 10
                    }
                    remote-mep-auto-discovery {
                        admin-state enable
                        aging-timer 86400
                    }
                    association-meps 1 {
                    }
```

```
                    association-meps 2 {
                    }
                    mep 1 {
                        admin-state enable
                        direction down
                        ccm-ltm-priority 1
                        interface-ref {
                            interface ethernet-1/10
                            subinterface 3
                        }
                        continuity-check {
                            ccm-transmit enable
                            ccm-local-fault {
                                action permit
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Configuring ETH-CCM (7730 SXR devices)

The following example enables the Ethernet continuity check and configures the ETH-CCM parameters.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm domain MD1 association MA1
    oam {
        ethcfm {
            domain MD1 {
                association MA1 {
                    association-format string
                    sender-id-permission-type chassis
                    ccm-interval 10ms
                    ma-name {
                        name association1
                    }
                    network-instance {
                        name test
                    }
                    ccm-hold-time {
                        delay-timeout 10
                    }
                    remote-mep-auto-discovery {
                        admin-state enable
                        aging-timer 86400
                    }
                    association-meps 1 {
                    }
                    association-meps 2 {
                    }
                    mep 1 {
                        admin-state enable
                        direction down
                        ccm-ltm-priority 1
                        interface-ref {
                            interface ethernet-1/10
                            subinterface 3
                        }
                        continuity-check {
                            ccm-transmit enable
```

```
                            ccm-local-fault {
                                action permit
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Configuring chassis type as local for sender ID TLV

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm sender-id
    oam {
        ethcfm {
            sender-id {
                chassis-type local
                chassis-local-name Nokia123
            }
        }
    }
```

Example: Configuring chassis type as system for sender ID TLV

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm sender-id
    oam {
        ethcfm {
            sender-id {
                chassis-type system
            }
        }
    }
```

4.3.5.6 Configuring MAC address allocation modes

Procedure

You can configure MAC address allocation mode using the **oam ethcfm mac-allocation mode** command. The MAC address allocation mode, **any** applies only to 7730 SXR platforms.

See Configuring MAC address mode in 7250 IXR platforms for an example of a MAC address mode configuration in 7250 IXR platforms.

See Configuring MAC address mode in 7730 SXR platforms for an example of a MAC address mode configuration in 7730 SXR platforms.

You can then assign the MAC address allocation to a MEP as shown in the example, Assigning the mac-allocation to a MEP.

Example: Configuring MAC address mode in 7250 IXR platforms

This example configures MAC address mode in 7250 IXR platforms.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm mac-allocation
    oam {
        ethcfm {
```

```
            mac-allocation {
                mode mac-pool
                custom-mac-pool pool1 {
                    starting-mac 00:00:00:00:00:FE
                    count 1
                }
            }
        }
    }
```

Example: Configuring MAC address mode in 7730 SXR platforms

This example configures MAC address mode in 7730 SXR platforms.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm mac-allocation
    oam {
        ethcfm {
            mac-allocation {
                mode any
                custom-mac-pool pool1 {
                    starting-mac 00:00:00:00:00:FE
                    count 64
                }
            }
        }
    }
```

Example: Assigning the mac-allocation to a MEP

This example assigns the MAC address allocation to a MEP.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ethcfm domain md-1
    oam {
        ethcfm {
            domain md-1 {
                domain-format none
                association as-1 {
                    association-format string
                    ma-name {
                        name name-1
                    }
                    network-instance {
                        name n-1
                    }
                    association-meps 1 {
                    }
                    mep 1 {
                        admin-state enable
                        direction down
                        interface-ref {
                            interface ethernet-1/1
                            subinterface 1
                        }
                        mac-address {
                            custom-mac-pool {
                                name pool1
                                index 1
                            }
                        }
                    }
                }
```

```
            }
        }
    }
```

4.3.5.7 Performing an Ethernet CFM loopback test

Procedure

To initiate an ETH-CFM loopback test, execute the **tools oam ethcfm domain association mep on-demand loopback** command, specifying the domain, association, MEP, and test parameters such as target MAC address/remote MEP ID. See Initiating ETH-CFM linktrace test using MAC address and Initiating ETH-CFM linktrace test using remote MEP ID for examples of ETH-CFM loopback test initiation using MAC address and remote MEP ID respectively.

To display the results, use the **info from state oam ethcfm domain association mep loopback** command, specifying the domain, association, and MEP. See Displaying ETH-CFM loopback test results for an example of an ETH-CFM loopback test result.

To terminate an ETH-CFM loopback test, execute the **tools oam ethcfm domain association mep terminate-active-test loopback**, specifying the domain, association, and MEP information. See Terminating ETH-CFM loopback test for an example of an ETH-CFM loopback test termination.

Example: Initiating an ETH-CFM loopback test using MAC address

The following example initiates the ETH-CFM loopback test using MAC address.

```
--{ + candidate shared default }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 11 on-demand loopback target 00:01:01:FF:00:1F
/oam/ethcfm/domain[domain-id=MD1]/association[association-id=MA1]/mep[mep-id=11]:
    Loopback Test from '.oam.ethcfm.domain{.domain_id=="MD1"}.association{.association_id==
"MA1"}.mep{.mep_id==11}' to 00:01:01:FF:00:1F is initiated.
```

Example: Initiating ETH-CFM loopback test using remote MEP ID

The following example initiates the ETH-CFM loopback test using remote MEP ID.

```
--{ + candidate shared default }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 11 on-demand loopback target 10
/oam/ethcfm/domain[domain-id=MD1]/association[association-id=MA1]/mep[mep-id=11]:
    Loopback Test from '.oam.ethcfm.domain{.domain_id=="MD1"}.association{.association_id==
"MA1"}.mep{.mep_id==11}' to 00:01:01:FF:00:1F is initiated.
```

Example: Displaying ETH-CFM loopback test results

The following displays the ETH-CFM loopback test results.

```
--{ + running }--[  ]--
# info from state /oam ethcfm domain MD1 association MA1 mep 11 loopback
    oam {
        ethcfm {
            domain MD1 {
                association MA1 {
                    mep 11 {
                        loopback {
                            status inactive
                            next-sequence-number 31
                            unicast-latest-run {
                                test-status completed
```

```
                                start-time 2024-10-29T14:11:32.807Z
                                end-time 2024-10-29T14:11:34.470Z
                                remote-mep-id 10
                                destination-mac-address 00:01:01:FF:00:1F
                                priority 7
                                data-length 0
                                interval 0s
                                sequence-number 21
                                statistics {
                                    sent-packets 10
                                    received-in-order 10
                                    received-out-of-order 0
                                    received-unexpected-sequence-number 0
                                    received-bad-msdu 0
                                    packet-loss 0.00
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Terminating ETH-CFM loopback test

The following example terminates the ETH-CFM loopback test.

```
--{ + candidate shared default }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 11 terminate-active-test loopback
```

4.3.5.8 Performing an Ethernet CFM linktrace test

Procedure

To initiate an Ethernet CFM linktrace test, use the **tools oam ethcfm domain association mep on-demand linktrace** command, specifying the domain, association, MEP, and test parameters such as target MAC address/remote MEP ID and TTL value. See Initiating ETH-CFM linktrace test using MAC address and Initiating ETH-CFM linktrace test using remote MEP ID for examples of ETH-CFM linktrace test initiation using MAC address and MEP ID respectively.

To display the results, use the **info from state oam ethcfm domain association mep linktrace** command, specifying the domain, association, and MEP. See Displaying ETH-CFM linktrace test results for an example of an ETH-CFM linktrace test result.

To terminate an ETH-CFM linktrace test, execute the **tools oam ethcfm domain association mep terminate-active-test linktrace** command, specifying the domain, association, and MEP information. See Terminating ETH-CFM linktrace test for an example of an ETH-CFM linktrace test termination.

Example: Initiating ETH-CFM linktrace test using MAC address

The following example initiates the ETH-CFM linktrace test using MAC address.

```
--{ + candidate shared default }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 11 on-demand linktrace target 00:01:01:FF:00:1F
/oam/ethcfm/domain[domain-id=MD1]/association[association-id=MA1]/mep[mep-id=11]:
    Linktrace Test from '.oam.ethcfm.domain{.domain_id=="MD1"}.association{.association_id==
"MA1"}.mep{.mep_id==11}' to 00:01:01:FF:00:1F is initiated.
```

Example: Initiating ETH-CFM linktrace test using remote MEP ID

The following example initiates the ETH-CFM linktrace test using remote MEP ID.

```
--{ + candidate shared default }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 11 on-demand linktrace target 10
/oam/ethcfm/domain[domain-id=MD1]/association[association-id=MA1]/mep[mep-id=11]:
    Linktrace Test from '.oam.ethcfm.domain{.domain_id=="MD1"}.association{.association_id==
"MA1"}.mep{.mep_id==11}' to 00:01:01:FF:00:1F is initiated.
```

Example: Displaying ETH-CFM linktrace test results

```
--{ + running }--[  ]--
# info from state /oam ethcfm domain MD1 association MA1 mep 11 linktrace
    oam {
        ethcfm {
            domain MD1 {
                association MA1 {
                    mep 11 {
                        linktrace {
                            status inactive
                            next-transaction-number 1804289384
                            latest-run {
                                test-status completed
                                start-time 2024-10-29T14:13:29.987Z
                                end-time 2024-10-29T14:13:36.988Z
                                remote-mep-id 10
                                destination-mac-address 00:01:01:FF:00:1F
                                priority 7
                                transaction-id 1804289383
                                ttl 64
                                reply 1 {
                                    reply-ttl 63
                                    forwarded false
                                    terminal-mep true
                                    ltr-relay hit
                                    ingress-action ok
                                    ingress-mac 00:01:01:FF:00:1F
                                    last-egress-identifier {
                                        integer 0
                                        mac-address 00:01:03:FF:00:20
                                    }
                                    next-egress-identifier {
                                        integer 0
                                        mac-address 00:00:00:00:00:00
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Terminating ETH-CFM linktrace test

The following example terminates the ETH-CFM linktrace test.

```
--{ + candidate shared default }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 11 terminate-active-test linktrace
```

4.3.5.9 Displaying ETH-CFM statistics

Procedure

To display system-level statistics, use the **info from state oam ethcfm statistics** command. See Displaying system-level ETH-CFM statistics for an example of ETH-CFM statistics at the system level.

To display MEP-level statistics, use the **info from state oam ethcfm domain association mep** command, specifying the domain name, association name, and MEP ID. Per OpCode statistics are only available on the 7730 SXR. See Displaying MEP-level ETH-CFM statistics for an example of ETH-CFM statistics at the MEP level.

Example: Displaying system-level ETH-CFM statistics

The following example displays the ETH-CFM statistics at the system level (7730 SXR example).

```
--{ + candidate shared default }--[  ]--
# info with-context from state oam ethcfm statistics
    oam {
        ethcfm {
            statistics {
                receive-count 3
                transmit-count 3
                receive-congestion-drops 0
                transmit-congestion-drops 0
                error-discards 0
                opcode total {
                    transmitted 2598
                    received 2598
                }
                opcode other {
                    transmitted 0
                    received 0
                }
                opcode ccm {
                    transmitted 2595
                    received 2595
                }
                opcode lbr {
                    transmitted 0
                    received 3
                }
                opcode lbm {
                    transmitted 3
                    received 0
                }
                opcode ltr {
                    transmitted 0
                    received 1
                }
                opcode ltm {
                    transmitted 1
                    received 0
                }
                opcode dmr {
                    transmitted 0
                    received 0
                }
                opcode dmm {
                    transmitted 0
                    received 0
```

```
                }
                opcode slr {
                    transmitted 0
                    received 0
                }
                opcode slm {
                    transmitted 0
                    received 0
                 }
            }
        }
    }
```

Example: Displaying MEP-level ETH-CFM statistics

The following example displays the ETH-CFM statistics at the MEP level.

```
--{ + candidate shared default }--[  ]--
# info from state oam ethcfm domain MD1 association MA1 mep 30
oam {
    ethcfm {
            domain MD1 {
                association MA1 {
                    mep 30 {
                        admin-state enable
                        direction down
                        mac-address AA:BB:CC:AC:AC:AC
                        ccm-ltm-priority 7
                        interface-ref {
                            interface lag1
                            subinterface 2
                        }
                        continuity-check {
                            ccm-transmit enable
                            lowest-fault-priority-defect mac-remote-error-xcon
                            highest-priority-defect-found none
                            ccm-sequence-error-count 0
                            sent-interface-status up
                            sent-port-status up
                            sent-remote-defect-indicator false
                            active-defects [
                                none
                            ]
                            ccm-local-fault {
                                action deny
                            }
                        }
                        remote-mep 40 {
                            mac-address AA:BB:CC:CA:CA:CA
                            auto-discovered false
                            receiving-ccm true
                            remote-mep-state ok
                            remote-mep-failed-ok-time 374
                            remote-defect-indicator false
                            port-status-tlv up
                            interface-status-tlv up
                        }
                        loopback {
                            status inactive
                            next-sequence-number 4
                            unicast-latest-run {
                                test-status completed
                                start-time 2024-04-30T11:11:46.483Z
```

```
                        end-time 2024-04-30T11:11:49.498Z
                        remote-mep-id 0
                        destination-mac-address AA:BB:CC:CA:CA:CA
                        priority 7
                        data-length 0
                        interval 1s
                        sequence-number 1
                        statistics {
                            sent-packets 10
                            received-in-order 10
                            received-out-of-order 0
                            received-bad-msdu 0
                            packet-loss 0.00
                        }
                    }
                }
                linktrace {
                    status inactive
                    next-transaction-number 1804289384
                    latest-run {
                        test-status completed
                        start-time 2024-04-30T11:11:53.877Z
                        end-time 2024-04-30T11:12:00.878Z
                        remote-mep-id 0
                        destination-mac-address AA:BB:CC:CA:CA:CA
                        priority 7
                        transaction-id 1804289383
                        ttl 64
                        reply 1 {
                            reply-ttl 63
                            forwarded false
                            terminal-mep true
                            ltr-relay hit
                            egress-action ok
                            egress-mac AA:BB:CC:CA:CA:CA
                            last-egress-identifier {
                                integer 0
                                mac-address AC:AC:AC:CC:BB:AA
                            }
                            next-egress-identifier {
                                integer 0
                                mac-address 00:00:00:00:00:00
                            }
                        }
                    }
                }
                opcode total {
                    transmitted 2598
                    received 2598
                }
                opcode other {
                    transmitted 0
                    received 0
                }
                opcode ccm {
                    transmitted 2595
                    received 2595
                }
                opcode lbr {
                    transmitted 0
                    received 3
                }
                opcode lbm {
                    transmitted 3
```

```
                        received 0
                    }
                    opcode ltr {
                        transmitted 0
                        received 1
                    }
                    opcode ltm {
                        transmitted 1
                        received 0
                    }
                    opcode dmr {
                        transmitted 0
                        received 0
                    }
                    opcode dmm {
                        transmitted 0
                        received 0
                    }
                    opcode slr {
                        transmitted 0
                        received 0
                    }
                    opcode slm {
                        transmitted 0
                        received 0
                    }
                }
            }
        }
    }
}
```

4.3.5.10 Clearing the learned remote MEP MAC address

Procedure

You can clear the automatically learned remote MEP MAC address from the Layer 2 address resolution mapping table at multiple levels of the hierarchy.

Example: Clearing learned remote MEP MAC addresses at system-level

The following example clears automatically learned remote MEP MAC addresses at the system level.

```
--{ + running }--[  ]--
# tools oam ethcfm delete-learned-remote-macs
```

Example: Clearing learned remote MEP MAC addresses at domain-level

The following example clears automatically learned remote MEP MAC addresses at the domain level.

```
--{ + running }--[  ]--
# tools oam ethcfm domain MD1 delete-learned-remote-macs
```

Example: Clearing learned remote MEP MAC addresses at association-level

The following example clears automatically learned remote MEP MAC addresses at the association level.

```
--{ + running }--[  ]--
```

```
# tools oam ethcfm domain MD1 association MA1 delete-learned-remote-macs
```

Example: Clearing learned remote MEP MAC addresses at MEP-level

The following example clears automatically learned remote MEP MAC addresses at the MEP level.

```
--{ + running }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 1 delete-learned-remote-macs
```

Example: Clearing learned remote MEP MAC addresses at remote MEP-level

The following example clears automatically learned remote MEP MAC addresses at the remote MEP level.

```
--{ + running }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 1  remote-mep 2 delete-learned-remote-
macs
```

4.3.5.11 Clearing automatically learned MEPs

Procedure

You can clear automatically learned remote MEPs from the remote MEP database, including all its attributes and at multiple levels of the hierarchy.

Example: Clearing automatically learned MEPs at system-level

The following example clears automatically learned remote MEPs at the system level.

```
--{ + running }--[  ]--
# tools oam ethcfm delete-auto-discovered-meps
```

Example: Clearing automatically learned MEPs at domain-level

The following example clears automatically learned remote MEPs at the domain level.

```
--{ + running }--[  ]--
# tools oam ethcfm domain MD1 delete-auto-discovered-meps
```

Example: Clearing automatically learned MEPs at association-level

The following example clears automatically learned remote MEPs at the association level.

```
--{ + running }--[  ]--
# tools oam ethcfm domain MD1 association MA1 delete-auto-discovered-meps
```

Example: Clearing automatically learned MEPs at remote MEP-level

The following example clears automatically learned remote MEPs at the MEP level.

```
--{ + running }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 1 remote-mep 2 delete-auto-discovered-
meps
```

4.3.5.12 Clearing ETH-CFM system statistics

Procedure

You can clear the ETH-CFM system statistics under the **ethcfm statistics** command that resets the benchmark in the application.

Example: Clearing ETH-CFM system statistics

The following example clears the ETH-CFM system statistics.

```
--{ running }--[  ]--
# tools oam ethcfm clear-cfm-statistics
```

4.3.5.13 Clearing ETH-CFM statistics for each MEP

Procedure

You can clear the ETH-CFM statistics for each MEP under the **ethcfm domain association mep-id** command that resets the benchmark in the application.

Example: Clearing ETH-CFM statistics for each MEP

The following example clears statistics for each MEP.

```
--{ + running }--[  ]--
# tools oam ethcfm domain MD1 association MA1 mep 1 clear-cfm-statistics
```

4.4 Bidirectional Forwarding Detection

Note: This feature is supported on 7250 IXR, 7220 IXR, and 7730 SXR platforms. 7730 SXR supports hardware-based BFD sessions. 7250 IXR and 7220 IXR support software-based BFD.

BFD is a lightweight mechanism used to monitor the liveliness of a remote neighbor. It is lightweight enough so that the ongoing sending and receiving mechanism can be implemented in the forwarding hardware. Because of its lightweight nature, BFD can send and receive messages at a much higher rate than other control plane hello mechanisms, providing faster detection of connection failures.

SR Linux supports BFD asynchronous mode, where BFD control packets are sent between two systems to activate and maintain BFD neighbor sessions between them.

BFD can be configured to monitor connectivity for the following:

- BGP peers – see Configuring BFD under the BGP protocol
- next hops for static routes – see Configuring BFD for static routes
- OSPF adjacencies – see Configuring BFD under OSPF
- IS-IS adjacencies – see Configuring BFD under IS-IS
- link layer LDP adjacencies - see Configuring BFD on an LDP interface

SR Linux supports one BFD session per port/connector, or up to 1152 sessions for an eight slot chassis, depending on the hardware configuration.

On SR Linux systems that support link aggregation groups (LAGs), SR Linux supports micro-BFD, where BFD sessions are established for individual members of a LAG. If the BFD session for one of the links indicates a connection failure, the link is taken out of service from the perspective of the LAG. See Micro-BFD.

4.4.1 BFD control packet

The base BFD specification does not specify the encapsulation type to be used for sending BFD control packets. Instead, use the appropriate encapsulation type for the medium and network. The encapsulation for BFD over IPv4 and IPv6 networks is specified in RFC 5881, *Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)*, and RFC 5883, *Bidirectional Forwarding Detection (BFD) for Multihop Paths*, BFD for IPv4 and IPv6. This specification requires that BFD control packets be sent over UDP with a destination port number of 3784 (single hop) or 4784 (multihop paths) and the source port number must be within the range 49152 to 65535.

Also, the TTL of all transmitted BFD packets must have an IP TTL of 255. All BFD packets received must have an IP TTL of 255 if authentication is not enabled. If authentication is enabled, the IP TTL should be 255, but can still be processed if it is not (assuming the packet passes the enabled authentication mechanism).

If multiple BFD sessions exist between two nodes, the BFD discriminator is used to de-multiplex the BFD control packet to the appropriate BFD session.

4.4.2 Control packet format

The BFD control packet has two sections: a mandatory section and an optional authentication section.

Figure 6: Mandatory frame format

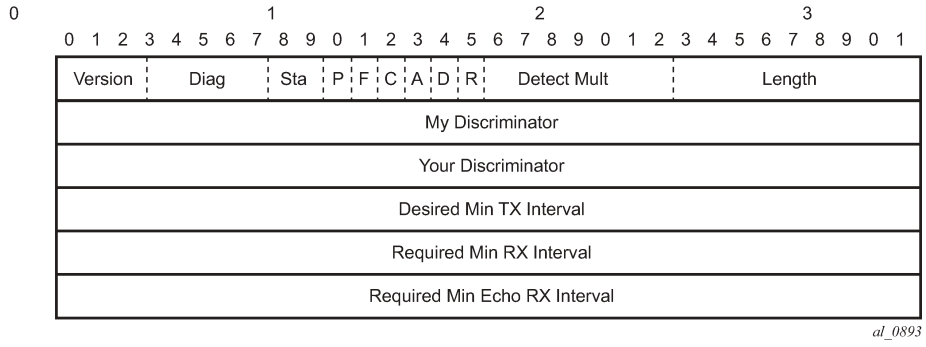


Table 9: BFD control packet field descriptions

Field	Description
Vers	The version number of the protocol. The initial protocol version is 0.
Diag	A diagnostic code specifying the local system's reason for the last transition of the session from Up to some other state.

Field	Description
	Possible values are: 0-no diagnostic 1-control detection time expired 2-echo function failed 3-neighbor signaled session down 4-forwarding plane reset 5-path down 6-concatenated path down 7-administratively down
D Bit	The demand mode bit. (Not supported)
P Bit	The poll bit. If set, the transmitting system is requesting verification of connectivity, or of an option change.
F Bit	The final bit. If set, the transmitting system is responding to a received BFD control packet that had the poll (P) bit set.
Rsvd	Reserved bits. These bits must be zero on transmit and ignored on receipt.
Length	Length of the BFD control packet, in bytes.
My Discriminator	A unique, non-zero discriminator value generated by the transmitting system, used to demultiplex multiple BFD sessions between the same pair of systems.
Your Discriminator	The discriminator received from the corresponding remote system. This field reflects back the received value of my discriminator, or is zero if that value is unknown.
Desired Min TX Interval	This is the minimum interval, in microseconds, that the local system would like to use when transmitting BFD control packets.
Required Min RX Interval	This is the minimum interval, in microseconds, between received BFD control packets that this system is capable of supporting.
Required Min Echo RX Interval	This is the minimum interval, in microseconds, between received BFD echo packets that this system is capable of supporting. If this value is zero, the transmitting system does not support the receipt of BFD echo packets.

4.4.3 Configuring BFD for a subinterface

Procedure

You can enable BFD with an associated subinterface and set values for intervals and criteria for declaring a session down.

Timer values are in microseconds. The detection interval for the BFD session is calculated by multiplying the value of the negotiated transmission interval by the value specified in this field.

Example

The following example configures BFD for a subinterface.

```
--{ + candidate shared default }--[  ]--
# info with-context bfd
    bfd {
        subinterface ethernet-1/2.1 {
            admin-state enable
            desired-minimum-transmit-interval 250000
            required-minimum-receive 250000
            detection-multiplier 3
            minimum-echo-receive-interval 0
            max-hop-count 2
        }
    }
```

4.4.4 Configuring BFD under the BGP protocol

Procedure

You can configure BFD under the BGP protocol at the global, group, or neighbor level.

Before enabling BFD, you must first configure it for a subinterface and set timer values. See Configuring BFD for a subinterface.

Example: Configure BFD under the BGP protocol at the global level

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default
    network-instance default {
        protocols {
            bgp {
                autonomous-system 65002
                router-id 2.2.2.2
                failure-detection {
                    enable-bfd true
                }
                afi-safi ipv4-unicast {
                    admin-state enable
                }
            }
        }
    }
```

Example: Configure BFD for a BGP peer group

The following example configures BFD for the links between peers within an associated BGP peer group.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default protocols bgp group test
    network-instance default {
        protocols {
            bgp {
                group test {
                    failure-detection {
                        enable-bfd true
                    }
```

```
                }
            }
        }
    }
```

Example: Configure BFD for BGP neighbors

The following example configures BFD for the link between BGP neighbors.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default protocols bgp neighbor 192.168.0.1
    network-instance default {
        protocols {
            bgp {
                neighbor 192.168.0.1 {
                    peer-group test
                    failure-detection {
                        enable-bfd true
                    }
                    afi-safi ipv4-unicast {
                        admin-state enable
                    }
                }
            }
        }
    }
```

4.4.5 Configuring BFD for static routes

Procedure

You can use BFD as a failure detection mechanism for monitoring the reachability of next hops for static routes. When BFD is enabled for a static route, it makes an active BFD session between the local router and the defined next hops required as a condition for a static route to be operationally active.

You enable BFD for specific next-hop groups; as a result, BFD is enabled for any static route that refers to the next-hop group. If multiple next hops are defined within the next-hop group, a BFD session is established between the local address and each next hop in the next-hop group.

A static route is considered operationally up if at least one of the configured next-hop addresses can establish a BFD session. If the BFD session fails, the associated next hop is removed from the FIB as an active next hop.

Example

The following example enables BFD for a static route next hop:

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default next-hop-groups
    network-instance default {
        next-hop-groups {
            group static-ipv4-grp {
                admin-state enable
                nexthop 1 {
                    failure-detection {
                        enable-bfd {
                            local-address 192.0.2.1
                        }
                    }
```

```
                }
            }
        }
    }
```

A BFD session is established between the address configured with the **local-address** parameter and each next-hop address before that next-hop address is installed in the forwarding table.

All next-hop BFD sessions share the same timer settings, which are taken from the BFD configuration for the subinterface where the address in **local-address** parameter is configured. See Bidirectional Forwarding Detection.

4.4.6 Configuring BFD under OSPF

Procedure

For OSPF and OSPFv2, you can enable BFD at the interface level to monitor the connectivity between the router and its attached network.

Example: Configuring BFD under OSPF

The following example configures BFD under the OSPF protocol:

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default protocols ospf
    network-instance default {
        protocols {
            ospf {
                instance o1 {
                    version ospf-v2
                    router-id 2.2.2.2
                    area 1.1.1.1 {
                        interface ethernet-1/12.1 {
                            failure-detection {
                                enable-bfd true
                            }
                        }
                    }
                }
            }
        }
    }
```

4.4.7 Configuring BFD under IS-IS

Procedure

You can configure BFD at the interface level for IS-IS. You can optionally configure a BFD-enabled TLV to be included for IPv4 or IPv6 on the IS-IS interface.

Example: Configuring BFD under IS-IS

The following example configures BFD under the IS-IS protocol:

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default protocols isis
    network-instance default {
```

```
        protocols {
            isis {
                instance i1 {
                    ipv4-unicast {
                        admin-state enable
                    }
                    interface ethernet-1/12.1 {
                        ipv4-unicast {
                            admin-state enable
                            enable-bfd true
                            include-bfd-tlv true
                        }
                    }
                }
            }
        }
    }
```

4.4.8 Configuring BFD on an LDP interface

Procedure

You can configure BFD on an IPv4 or IPv6 LDP interface.

Example: Configuring BFD on an LDP interface

This example enables BFD on an LDP interface.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default protocols ldp
    network-instance default {
        protocols {
            ldp {
                dynamic-label-block d1
                discovery {
                    interfaces {
                        hello-holdtime 30
                        hello-interval 10
                        interface ethernet-1/12.1 {
                            ipv4 {
                                admin-state enable
                                enable-bfd true
                            }
                            ipv6 {
                                admin-state enable
                                enable-bfd true
                            }
                        }
                    }
                }
            }
        }
    }
```

4.4.9 Viewing the BFD state

Procedure

Use the **info from state** command to verify the BFD state for a network-instance.

Example: Viewing the BFD state

```
# info with-context from state bfd network-instance default peer 30
    bfd {
        network-instance default {
            peer 30 {
                oper-state up
                local-address 192.168.1.5
                remote-address 192.168.1.3
                remote-discriminator 25
                subscribed-protocols bgp_mgr
                session-state UP
                remote-session-state UP
                last-state-transition 2020-01-24T16:22:55.224Z
                failure-transitions 0
                local-diagnostic-code NO_DIAGNOSTIC
                remote-diagnostic-code NO_DIAGNOSTIC
                remote-minimum-receive-interval 1000000
                remote-control-plane-independent false
                active-transmit-interval 250000
                active-receive-interval 250000
                remote-multiplier 3
                async {
                    last-packet-transmitted 2020-01-24T16:23:19.385Z
                    last-packet-received 2020-01-24T16:23:18.906Z
                    transmitted-packets 32
                    received-packets 32
                    up-transitions 1
                }
            }
        }
    }
```

4.5 Micro-BFD

Note: Micro-BFD is supported on all SR Linux systems that also support LAGs: 7250 IXR, 7220 IXR, and 7730 SXR.

Micro-BFD refers to running BFD over the individual links in a LAG to monitor the bidirectional liveliness of the Ethernet links that make up the LAG.

A LAG member cannot be made operational within the LAG until the micro-BFD session is fully established. If a micro-BFD session fails, the corresponding Ethernet link is taken out of service from the perspective of the LAG.

Micro-BFD is supported on Ethernet LAG interfaces with an IP interface. Micro-BFD sessions are associated with each individual link. When enabled, the state of the individual links depends on the micro-BFD session state:

- Micro-BFD sessions must be established between both endpoints of a link before the link can be operationally up.

- If the micro-BFD session fails, the associated Ethernet link becomes operationally down from the perspective of the LAG.
- If LACP is not enabled for the LAG and the Ethernet port is up, the system attempts to re-establish the micro-BFD session with the far end of the link.
- If LACP enabled for the LAG and the Ethernet port is up, the system attempts to re-establish the micro-BFD session with the far end of the link when LACP reaches distributing state.

If a link is not active for forwarding from the perspective of a LAG, ARP can still be performed across the link. For example, when a link is being brought up, and its micro-BFD session is not yet established, ARP can still be performed for the MAC address at the far end of the link, even though the link is not yet part of the LAG.

Micro-BFD packets bypass ingress and egress subinterface/interface ACLs, but received micro-BFD packets can be matched by CPM filters for filtering and logging.

4.5.1 Configuring micro-BFD for a LAG interface

Procedure

To configure micro-BFD for a LAG interface, you configure IP addresses to be used as the source address for IP packets and a remote address for the far end of the BFD session.

You can specify the minimum interval in microseconds between transmission of BFD control packets, as well as the minimum acceptable interval between received BFD control packets. The detection-multiplier setting specifies the number of packets that must be missed to declare the BFD session as down.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context bfd micro-bfd-sessions
    bfd {
        micro-bfd-sessions {
            lag-interface lag1 {
                admin-state enable
                local-address 192.35.2.5
                remote-address 192.35.2.3
                desired-minimum-transmit-interval 250000
                required-minimum-receive 250000
                detection-multiplier 3
            }
        }
    }
```

4.5.2 Viewing the micro-BFD state

Procedure

Use the **info from state** command to verify the micro-BFD state for members of a LAG interface.

Example

```
# info from state with-context bfd micro-bfd-sessions lag-interface lag1 member-interface
 ethernet 2/1
    micro-bfd-sessions
        lag-interface lag1 {
```

```
            admin-state UP
            local-address 192.0.2.5
            remote-address 192.0.2.3
            desired-minimum-transmit-interval 250000
            required-minimum-receive 250000
            detection-multiplier 3
            member-interface ethernet 2/1 {
                session-state UP
                remote-session-state UP
                last-state-transition 2020-01-24T16:22:55.224Z
                last-failure-time 2020-01-24T16:22:55.224Z
                failure-transitions 0
                local-discriminator 25
                remote-discriminator 25
                local-diagnostic-code NO_DIAGNOSTIC
                remote-diagnostic-code NO_DIAGNOSTIC
                remote-minimum-receive-interval 1000000
                remote-control-plane-independent false
                active-transmit-interval 250000
                active-receive-interval 250000
                remote-multiplier 3
                async {
                   last-clear 2020-01-23T16:21:19.385Z
                   last-packet-transmitted 2020-01-24T16:23:19.385Z
                   last-packet-received 2020-01-24T16:23:18.906Z
                   transmitted-packets 32
                   received-errored-packets 3
                   received-packets 32
                   up-transitions 1
                }
            }
        }
    }
```

4.6 Seamless Bidirectional Forwarding Detection (S-BFD)

Note: This feature is supported on 7250 IXR, 7220 IXR, and 7730 SXR platforms.

BFD detects connection failures faster than other hello mechanisms. However, if many BFD sessions are configured to detect links, very long negotiation times result in reduced system performance. You can configure seamless bidirectional forwarding detection (S-BFD), which is a simplified mechanism that speeds up a BFD session by eliminating the negotiation and state establishment process. This is accomplished primarily by predetermining the session discriminator and using specific mechanisms to distribute the discriminators to a remote network entity. This allows client applications or protocols to quickly initiate and perform connectivity tests. A per-session state is maintained only at the head-end of a session. The tail-end reflects the BFD control packets back to the head-end.

4.6.1 Initiator and reflector

An S-BFD session is established between an initiator and a reflector. SR Linux supports only one instance of a reflector in each node. A discriminator is assigned to initiator and reflector.

The initiator initiates an S-BFD session on a network node and performs a continuity test by sending S-BFD packets to the reflector. The reflector receives the S-BFD packet and reflects the S-BFD packet back along with the state value based on its current state.

The following information is swapped in the S-BFD response:

- The source and destination IP addresses
- The source and destination UDP ports
- The initiator and reflector discriminators

See Configuring an S-BFD reflector for information about how to configure a reflector. An SR Linux router can be both an initiator and a reflector, thereby allowing you to configure different S-BFD sessions.

4.6.2 S-BFD discriminator

SR Linux supports the following methods of mapping an S-BFD remote IP address with its discriminator:

- Static configuration
- Automatic learning using opaque IS-IS routing extensions

You can statically configure an S-BFD remote IP address and discriminator for each network instance. The S-BFD initiator immediately starts sending S-BFD packets if the discriminator value of the far-end reflector is known. A session set up is not required. The **INIT** state is not present in an S-BFD session. The initiator state changes from **AdminDown** to **Up** when it begins to send S-BFD packets. The following table lists the S-BFD packet information that the initiator sends to the reflector.

Table 10: Fields in S-BFD packet

Source IP address	This is the local session IP address. For IPv6, this is a global unicast address belonging to the node.
Destination IP address	This is the IP address of the reflector, and it needs to be configured.
My discriminator	This is the locally assigned discriminator.
Your discriminator	This is the discriminator value of the reflector, and it needs to be configured.

See Statically configuring an S-BFD discriminator for more information about how to configure an S-BFD discriminator.

If the initiator receives a valid response from the reflector with an **Up** state, the initiator declares the S-BFD session state as **Up**. If the initiator fails to receive a specific number of responses, as determined by the BFD multiplier in the BFD template for the session, the initiator declares the S-BFD session state as **Failed**. If any of the discriminators change, the session fails and the router attempts to restart with the new values. If the reflector discriminator changes at the far-end peer, the session fails. The mapping may not have been updated locally before the system checks for a new reflector discriminator from the local mapping table. Therefore the session is bounced and brought up with the new values. If any discriminator is deleted, the corresponding S-BFD sessions are deleted.

SR Linux supports automatic mapping of an S-BFD remote IP address with its discriminator using the IS-IS protocol extensions. The IS-IS protocol uses a sub-TLV of the capabilities TLV to advertise and distribute discriminators. See Automatically mapping an S-BFD discriminator for more information.

4.6.3 Routed and controlled return path

S-BFD supports the following forms of returning transmitted S-BFD packets back to the initiator:

- Routed return
- Controlled return path

In routed return, S-BFD uses an initiator-reflector model where an initiator sends S-BFD messages to a reflector using the discriminator of the reflectors. The reflector reflects the S-BFD message back to the initiator via IPv4 or IPv6 routing.

In controlled return path for SR-Policy, the initiating node embeds a SID, typically a binding SID that is used by the reflecting node, to determine the correct path back to the initiator. The S-BFD message is then forwarded to a path that is identical or similar to the original path that the message was sent by the initiator.

4.6.4 Seamless BFD for LSP monitoring

Note: This feature is supported on 7250 IXR and 7730 SXR platforms.

S-BFD is used to monitor the liveliness of TE-Policy Segment Lists via utilizing the same path with the same label furnishing as datapath traffic. In case of redundancy, Separate S-BFD sessions are established over both primary and backup paths. If an S-BFD session fails on the primary path, traffic switches to the backup path. If S-BFD sessions fail on both primary and backup paths, traffic is sent over an inactive backup path.

S-BFD is supported on segment lists for TE policies by binding a protection policy containing an S-BFD configuration to an imported TE policy. S-BFD packets are encapsulated on the TE policy segment lists and require the discriminator of the local node, as well as mappings to the far-end reflector node discriminators. BFD sets the remote discriminator at the initiator of the S-BFD session based on a lookup in the S-BFD reflector discriminator table, using the endpoint address of the TE policy candidate path. A candidate path of a TE policy is considered available only if the number of active S-BFD sessions equals or exceeds a configured threshold.

Protection policy

You can use the **system protection-policies policy** command to create a protection policy and configure the **revert-timer** and S-BFD parameters.

The **revert-timer** parameter controls when the system switches back to the primary or best path after it is recovered from a failure. By default, **revert-timer** is disabled. Depending on the TE policy type, **revert-timer** applies differently:

- For an uncolored TE policy, it applies to the primary segment list.
- For a colored TE policy, it applies to the best candidate path.

When a protection policy is applied to a TE policy, the **revert-timer** is updated according to the revert timer that is configured in the protection policy. If the protection policy is later removed from the TE policy, the **revert-timer** remains valid.

S-BFD parameters include:

- **mode**

- **threshold**
- **hold-down-timer**
- **wait-for-up-timer**
- **desired-minimum-transmit-interval**
- **detection-multiplier**

Protection modes

S-BFD can be configured to operate in the following protection modes:

- **monitored**
- **linear**
- **ecmp-protected**

The following table describes the behaviour of the protection modes in colored and uncolored TE policies.

Table 11: Protection modes

Protection mode	Uncolored TE policy	Colored TE policy
Monitored	**Setup** Setup is attempted only for the primary segment list. The standby segment list remains inactive and is not programmed.	Multiple segment lists are allowed for each candidate path. **Setup** Setup is attempted for all segment lists under the best candidate path (based on the candidate path **preference** value, with higher values first). S-BFD is applicable only to the best path. The S-BFD path requirement depends on the protection policy mode configuration. If no protection policy is assigned, the candidate path is attempted without S-BFD.
	Failure Failure of the primary segment list (because of S-BFD, SR-ISIS, or TE failure) initiates the setup of the next best available segment list, that is, either standby or secondary.	**Failure** Failure of all segment lists in the best candidate path initiates the setup of next best candidate path. An S-BFD failure on best candidate path does not trigger a failover. The next best candidate path is attempted only when the best path enters an invalid state (for example, unresolved first segment, BSID allocation failure, and so on).
	Revertive	**Revertive**

Protection mode	Uncolored TE policy	Colored TE policy
	When a primary segment list or a better segment list (a secondary segment list with a lower preference than the active one) becomes available, it is promoted to be the new active segment list. The S-BFD status for segment lists determines the active segment list within a specific TE-Policy. An S-BFD failure in the primary segment list triggers the activation of the standby segment list. If both the primary and standby segment lists fail, the system initiates the setup of a secondary segment list.	When a protection policy with a mode configuration is attached and the BSID matches that of the active path, the backup is promoted to be the active path in a make before break (MBB) fashion.
Linear	**Setup** The tunnel is set up by initially attempting all primary and standby segment lists. The two best candidate paths are set up and programmed as an active-backup pair.	A candidate-path can have only one segment list. If multiple segment lists are configured, only the one with the lowest index is programmed in the datapath. **Setup** The top three paths with the best preference are programmed. The first two paths are programmed with protection as an active/backup pair. The third path is programmed as a standby or inactive path. First path - active Second path - backup Third path - inactive
	Failure Failure of all of the primary and standby segment lists initiates the setup of secondary segment lists.	**Failure** S-BFD failure If the S-BFD of the first best path fails, failover between the first two candidate paths is triggered. If the S-BFD of both the first and second paths fail, the system switches to the already programmed third path, making it active. If the S-BFD of all the three selected paths fail, no additional candidate path is triggered until one of the first three paths become valid again.
	Revertive	**Revertive**

Protection mode	Uncolored TE policy	Colored TE policy
	When a better standby segment list becomes available, either based on a lower secondary segment list **preference** value or the primary segment list becomes available, it is promoted to be the active segment list. The S-BFD status for segment lists determines the active segment list of a specific TE-Policy. Both the active and backup segment lists (which can be standby or secondary) are programmed to the same protection group to enable seamless switchover or protection.	Whenever a better or the best candidate path becomes available again, it is promoted to the active state.
ECMP protected	Not applicable for uncolored policy.	Multiple segment lists are allowed for each candidate path. **Setup** Two candidate paths are configured, one as active and the other as standby or inactive backup mode. S-BFD runs on all segment lists. For both the active and standby or inactive candidate path, all segment lists are monitored through S-BFD.
		Failure A candidate path is declared down when the number of segment lists with S-BFD in the up state falls below a configured threshold. The standby candidate path is programmed but remains inactive. When the active candidate path is declared down, the standby candidate path is promoted to be the active path. **Active to standby control plane (CP) switchover** If the number of active segment lists on the active CP drops below a defined threshold and the standby CP has more segment lists than that threshold, the switchover is seamless as long as there is a protecting segment list with an established S-BFD session.

Protection mode	Uncolored TE policy	Colored TE policy
		In the event of a major failure such as a fiber cut or port failure affecting all segment lists, switchover is expected to complete in under 50ms after failure detection.
		Liveliness detection S-BFD is used to monitor the liveliness of each segment list within the candidate path.

Note: The only difference between monitored mode and linear mode is that, in monitored mode, only the active segment list is programmed in the datapath.

Threshold

The **threshold** setting specifies the minimum number of successful S-BFD sessions required to keep a TE policy path in the up state. If the number of up sessions falls below this value, the system marks the corresponding path as degraded. The default value is one. The **threshold** configuration applies only in **ecmp-protected** mode. In **linear** mode, an implicit threshold value is set to 1.

Example: S-BFD threshold

When two candidate paths are configured as follows:

- candidate path 1 has two valid segment lists
- candidate path 2 has six valid segment lists
- **ecmp-protection** mode is enabled with a preference order of candidate path 1 > candidate path 2
- **threshold** is set to 3

candidate path 1 remains down with the reason code **"operational segment lists below threshold"**. Candidate path 2 remains up as long as at least three of its S-BFD sessions are up.

Hold down timer

The hold down timer starts when the number of S-BFD sessions drops below the threshold. The TE policy path is not considered to be up again until the hold down timer expires and the number of S-BFD sessions meets or exceeds the threshold. Additionally, a grace period is provided after a session goes down to prevent BFD session flaps from impacting the active path.

Wait for up timer

The wait for up timer starts when S-BFD is first enabled on a segment list or when S-BFD transitions from up to down. If the timer expires and S-BFD has not yet come up, then the path is torn down by removing it from the Tunnel Table Manager (TTM) and the retry timer is started. The default value of the **wait-for-up-timer** is 4 seconds.

4.6.5 S-BFD state

S-BFD session state is reported at the network instance, policy, and system levels. See Viewing the S-BFD state for more information.

4.6.6 Statically configuring an S-BFD discriminator

Procedure

To statically map an S-BFD remote IP address with its discriminator for each network instance, you configure the **network-instance bfd seamless-bfd** command and specify the peer IP address and discriminator.

Example: Statically configuring an S-BFD discriminator

This is an example for statically configuring an S-BFD discriminator.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default bfd seamless-bfd
    network-instance default {
        bfd {
            seamless-bfd {
                peer 192.0.2.0 {
                    discriminator 524288
                }
            }
        }
    }
```

4.6.7 Automatically mapping an S-BFD discriminator

Procedure

SR Linux supports automatic mapping of an S-BFD remote IP address with its discriminator using IGP routing protocol extensions. The IS-IS protocol uses a sub-TLV of the capabilities TLV to distribute S-BFD discriminators. There is no explicit configuration to enable or disable router capability advertisement.

Example: Output from BFD state

This example shows an output of an automatically mapped S-BFD discriminator.

```
--{ + running }--[  ]--
# info from state with-context network-instance base protocols isis instance * level * link-state-
database lsp 0006.0006.0006.00-00 tlvs tlv router-capability router-capabilities capability 0 subtlvs
 subtlv rou
ter-capability-seamless-bfd-discriminator
    sbfd-discriminators {
        discriminator [
            524289
        ]
    }
```

4.6.8 Configuring an S-BFD reflector

Procedure

To enable and configure an S-BFD reflector, use the **network-instance bfd seamless-bfd reflector** command. You must allocate the discriminator value from the S-BFD reflector pool that ranges from 524288 to 526335.

Note:
Only a single reflector discriminator is supported for each network instance.

Example: Configuring an S-BFD reflector

The following example configures an S-BFD reflector.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default bfd seamless-bfd reflector abc
    network-instance default {
        bfd {
            seamless-bfd {
                reflector abc {
                    local-discriminator 524289
                    admin-state enable
                    description test
                }
            }
        }
    }
```

4.6.9 Configuring S-BFD in protection policy and associating TE policy

Procedure

Note: This feature is supported on 7250 IXR and 7730 SXR platforms.

You can configure S-BFD in the protection policy using the **system protection-policies policy** command and associate the protection policy to a TE policy using the **network-instance traffic-engineering-policies policy protection protection-policy** command.

Example: Configure S-BFD in a protection policy

This example configures S-BFD in a protection policy.

```
--{ + candidate shared default }--[  ]--
# info with-context system protection-policies policy protection-policy1
    system {
        protection-policies {
            policy protection-policy1 {
                revert-timer disable
                seamless-bfd {
                    detection-multiplier 3
                    desired-minimum-transmit-interval 1000000
                    hold-down-timer disable
                    wait-for-up-timer 4
```

```
                    mode linear
                    threshold 1
                }
            }
        }
    }
```

Example: Associating a protection policy to a TE policy

This example associates a protection policy to a TE policy.

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default traffic-engineering-policies policy te-1
    network-instance default {
        traffic-engineering-policies {
            policy te-1 {
                admin-state enable
                endpoint 10.0.0.1
                protection {
                    protection-policy protection-policy1
                }
            }
        }
    }
```

4.6.10 Viewing the S-BFD state

Procedure

Use the **info from state** command to verify the S-BFD state.

Example: Viewing the S-BFD state at network instance level

The following example displays the S-BFD status at the network instance level.

```
--{ + candidate shared default }--[  ]--
# info from state with-context network-instance default bfd seamless-bfd
    network-instance default {
        bfd {
            seamless-bfd {
                peer 192.0.2.0 {
                    discriminator 524288
                }
                reflector abc {
                    local-discriminator 524289
                    admin-state enable
                    description test
                }
            }
        }
    }
```

Example: Viewing the S-BFD state at the protection policy level

The following example displays the S-BFD status at the protection policy level.

```
--{ + candidate shared default }--[  ]--
# info from state with-context system protection-policies policy protection-policy1
    system {
```

```
        protection-policies {
            policy protection-policy1 {
                revert-timer disable
                seamless-bfd {
                    detection-multiplier 3
                    desired-minimum-transmit-interval 1000000
                    hold-down-timer disable
                    wait-for-up-timer 4
                    mode linear
                    threshold 1
                }
            }
        }
    }
```

Example: Viewing the S-BFD state at the system level

The following example displays the S-BFD status at the system level.

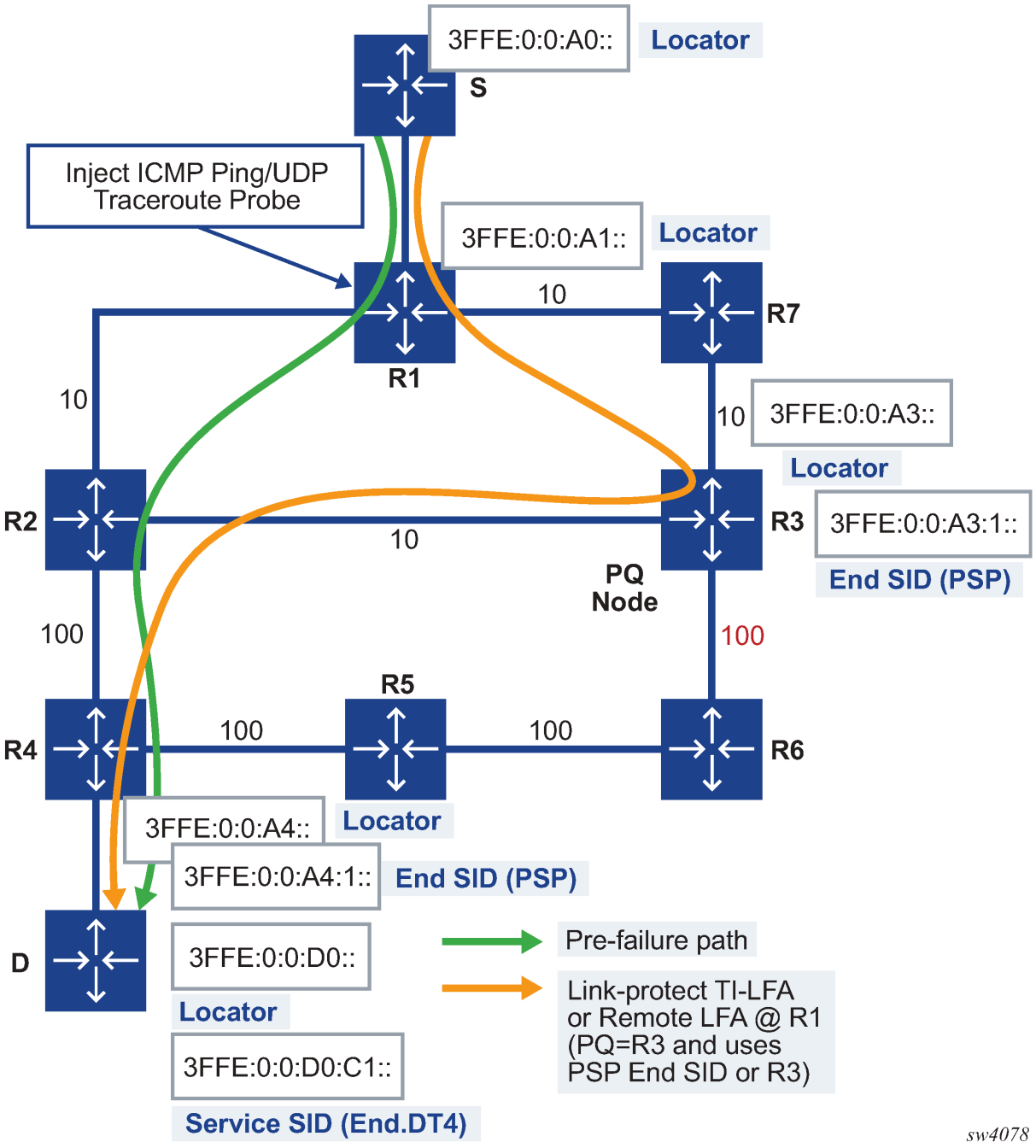
```
--{ + running }--[  ]--
# info from state with-context bfd
    bfd {
        total-bfd-sessions 2
        total-unmatched-bfd-packets 1
        network-instance base {
            peer 16385 {
                oper-state up
                local-address 1.1.1.3
                remote-address 127.0.64.1
                remote-discriminator 524289
                subscribed-protocols SRPOLICY
                session-state UP
                remote-session-state UP
                last-state-transition "2024-05-15T19:15:58.117Z (49 seconds ago)"
                failure-transitions 0
                local-diagnostic-code NO_DIAGNOSTIC
                remote-diagnostic-code NO_DIAGNOSTIC
                remote-minimum-receive-interval 1000000
                remote-control-plane-independent false
                active-transmit-interval 1000000
                active-receive-interval 1000000
                remote-multiplier 3
                te-policy-name C_to_Fipv4
                te-policy-segment-list-index 1
                te-policy-protocol-origin LOCAL
                te-policy-segment-list-lsp-index 216
                sr-policy-endpoint 1.1.1.6
                async {
                    last-packet-transmitted "2024-05-15T19:16:43.140Z (4 seconds ago)"
                    last-packet-received "2024-05-15T19:16:43.146Z (4 seconds ago)"
                    transmitted-packets 61
                    received-packets 61
                    up-transitions 1
                }
            }
            peer 16386 {
                oper-state up
                local-address 1.1.1.3
                remote-address 127.0.64.2
                remote-discriminator 524289
                subscribed-protocols SRPOLICY
                session-state UP
                remote-session-state UP
```

```
                last-state-transition "2024-05-15T19:15:58.119Z (49 seconds ago)"
                failure-transitions 0
                local-diagnostic-code NO_DIAGNOSTIC
                remote-diagnostic-code NO_DIAGNOSTIC
                remote-minimum-receive-interval 1000000
                remote-control-plane-independent false
                active-transmit-interval 1000000
                active-receive-interval 1000000
                remote-multiplier 3
                te-policy-name C_to_Fipv4
                te-policy-segment-list-index 2
                te-policy-protocol-origin LOCAL
                te-policy-segment-list-lsp-index 217
                sr-policy-endpoint 1.1.1.6
                async {
                    last-packet-transmitted "2024-05-15T19:16:43.651Z (4 seconds ago)"
                    last-packet-received "2024-05-15T19:16:43.695Z (4 seconds ago)"
                    transmitted-packets 62
                    received-packets 62
                    up-transitions 1
                }
            }
        }
    }
```

4.7 OAM support in Segment Routing IPv6 (SRv6)

The following figure shows an example configuration of Segment Routing using SRv6.

Figure 7: SRv6 OAM network setup



As shown in the preceding figure, the network administrator originates a ping or a traceroute probe on node R1 to test the path of an SRv6 locator of node D, an SRv6 segment identifier (SID) owned by node D, or an IP prefix resolved to an SRv6 tunnel toward node D. R1 is referred to as the sender node. Node D is referred to as the target node because it owns the target locator or SID being tested. A target node can be any router in the SRv6 network domain that either owns the target locator or SID, or a router at which

the OAM probe was extracted because a local route matches or because of the value of the hop-limit field setting in the packet.

The primary path to D is through R2 and R4. The link-protect TI-LFA backup path uses R3 as a PQ node, then continues through R2 and R4.

The **ping** and **traceroute** OAM commands are used to test an IPv6 prefix in a default network instance when resolved to an SRv6 tunnel.

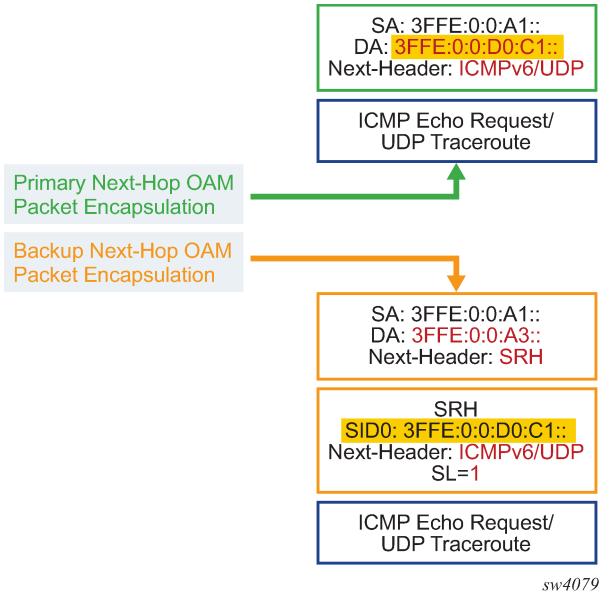
4.7.1 Ping or traceroute of SRv6 remote locator or remote SID (End, End.X, End.DT4, End.DT6, End.DT46, End.DX2, End.DT2M and End.DT2U)

The features in this section are in accordance with *draft-ietf-6man-spring-srv6-oam*, *Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)*.

4.7.1.1 Ingress PE router (sender node) behavior

The packet is encoded with a destination address set to the remote locator prefix or the specific remote SID, and the next-header field is ICMPv6 (for the ping Echo Request message) or UDP or TCP (for traceroute). The packet is encapsulated as shown in the following figure.

Figure 8: Packet encapsulation for ping or traceroute of a remote locator/SID



When the Topology-Independent Loop-Free Alternate (TI-LFA) or Remote LFA repair tunnel is activated, the LFA segment routing header (LFA SRH) is also pushed on the encapsulation of the SRv6 tunnel to the node D.

The outer IPv6 header hop-limit field is set according to the operation of the probe. For ping, the hop limit uses the default value of 254, or a user-entered value.

For traceroute, the hop limit is incrementally increased using one of the following:

- from 1 until the packet reaches the egress PE

- from the configured minimum value to the maximum value or until the packet reaches the egress PE

The ingress PE looks up the prefix of the locator or SID in the routing table and if a route exists, it forwards the packet to the next hop. The ingress PE does not check whether the target SID or locator was received in IS-IS or BGP.

4.7.1.2 Transit P router behavior

Ping and traceroute operate similarly to any data or OAM IPv6 packet when expiring (the value in the hop-limit field is equal to or less than 1) at a transit SRv6 node, regardless of whether this node is a SID termination. The datapath at the ingress network interface, where the packet is received, extracts the packet to the CPM. The CPM originates a TTL expiry ICMP reply message Type: "Time Exceeded", Code: "Time to Live exceeded in Transit".

The CPM sends the reply to the SRv6 router whose address is encoded in the SA field of the outer IPv6 header in the received packet. The source address is set to the system IPv6 address, if configured, or the address of the interface used to forward the packet to the next hop.

4.7.1.3 Egress PE router (target node) behavior

The datapath of the ingress network port in the destination router that owns the target SID extracts the packet to CPM. A traceroute packet is extracted based on the hop-limit field value of 1, before the route lookup. A ping packet is extracted after the route lookup matches a FIB entry of a local locator, End, or End.X SID.

The CPM checks that the target locator or SID address matches a local entry. This means that, the locator or SID is either configured manually by the user or auto-allocated by the locator module for use by IS-IS or BGP.

A match on the locator requires an exact match on the locator field, and both the function and argument fields must be zero. A match on a SID requires both the locator and function fields to match. The argument field is not checked.

When a match on a local locator or SID exists, the CPM replies with the following:

- **in the case of ICMPv6 ping**

  The CPM replies with an ICMPv6 Echo Reply message.

  The source address of the packet is set to the address in the DA field of the received echo request message.

- **in the case of UDP traceroute**

  The CPM replies with an ICMPv6 message (Type: "Destination unreachable", Code: "Port Unreachable").

  The source address is set to the system IPv6 address, if configured, or the address of the interface used to forward the packet to the next hop.

The following figures show the packet encapsulation from ingress PE to egress PE for both the primary path and the backup path. For the backup path, both the PSP and USP types of the LFA SRH are shown.

Figure 9: End-to-end packet encapsulation for ping or traceroute of a remote locator or SID (1)

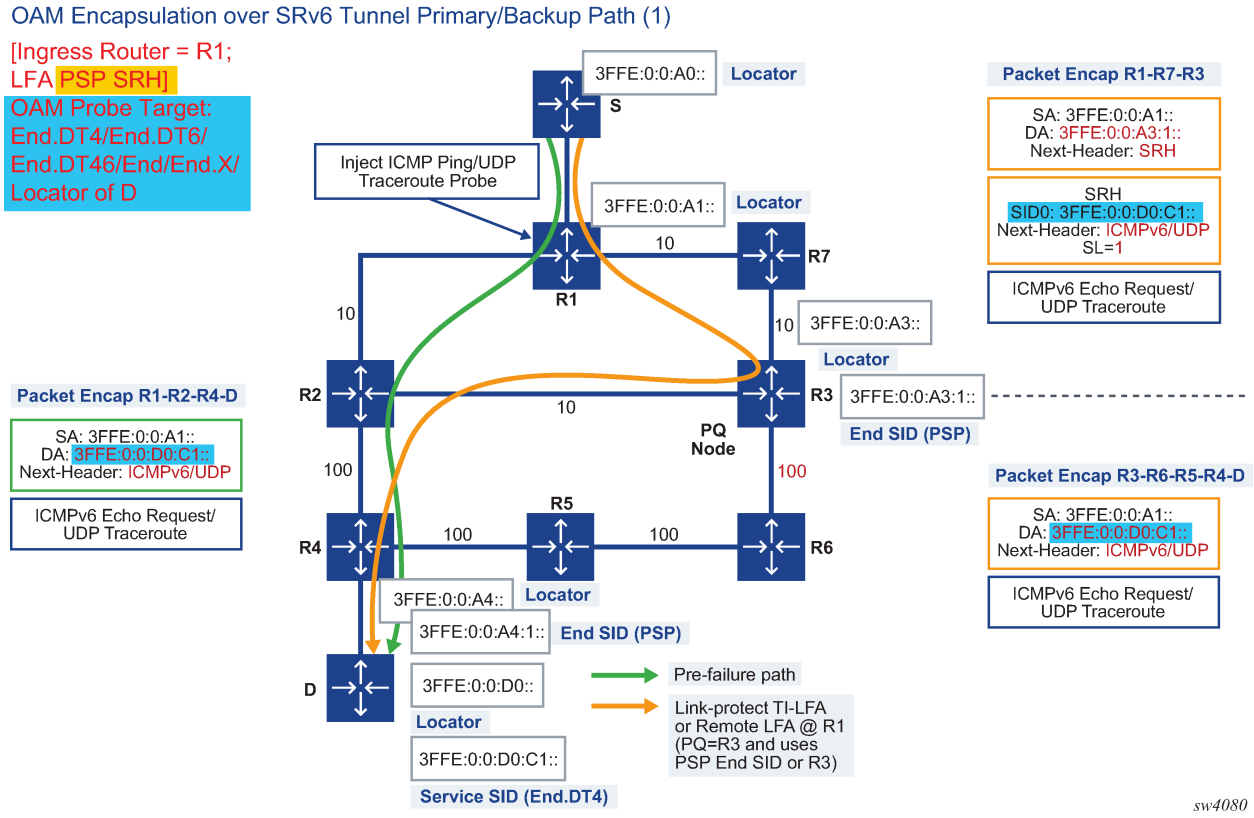
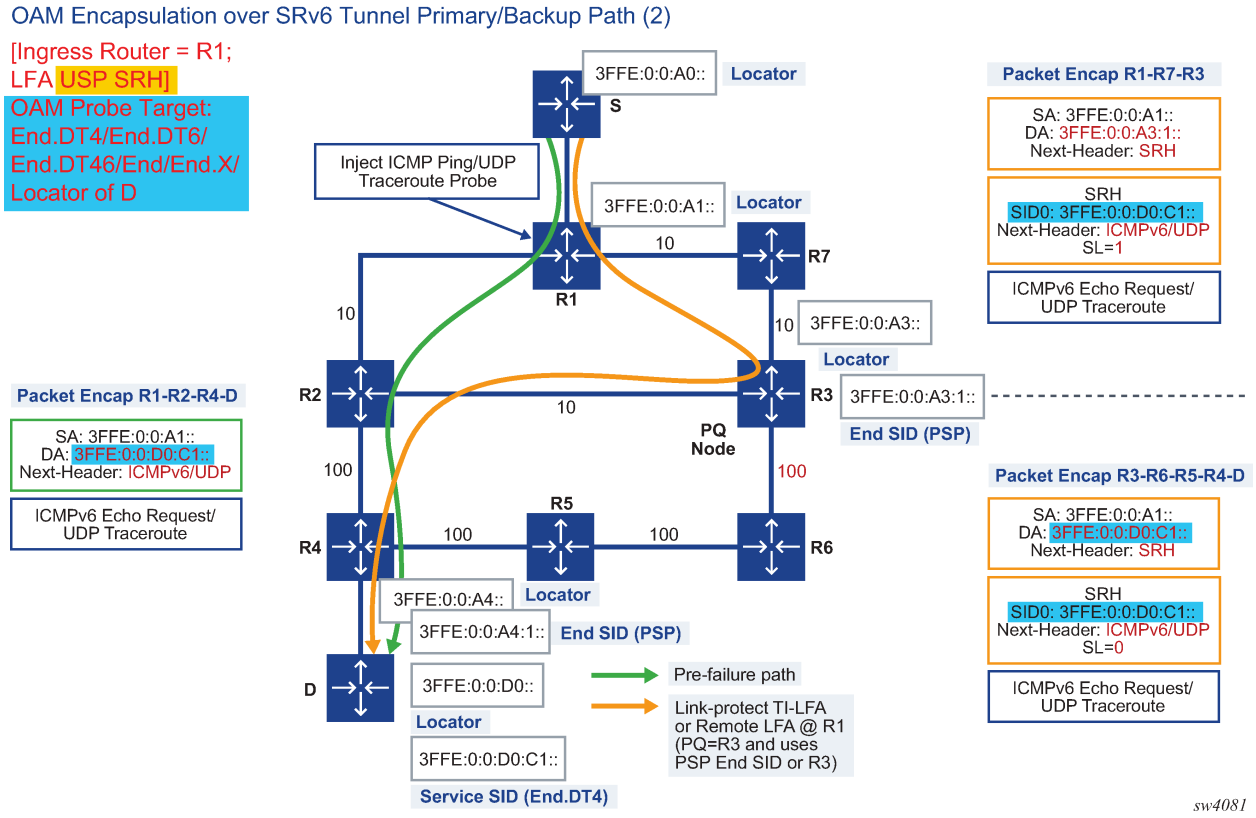


Figure 10: End-to-end packet encapsulation for ping or traceroute of a remote locator or SID (2)



4.7.2 Ping or traceroute of an IPv4 or IPv6 VRF prefix resolved to an SRv6 tunnel

This feature implements the existing behavior of a ping or a traceroute packet, originated at the ingress PE node, for a prefix resolved to an SRv6 tunnel. If the OAM ping or traceroute packet is received from the CE router and expires (hop-limit field value equal to or less than 1), the ingress PE node responds according to the current behavior. If the packet does not expire (hop-limit field value greater than 1), it is forwarded over the SRv6 tunnel as a datapath packet.

4.7.2.1 Ingress PE router (sender node) behavior

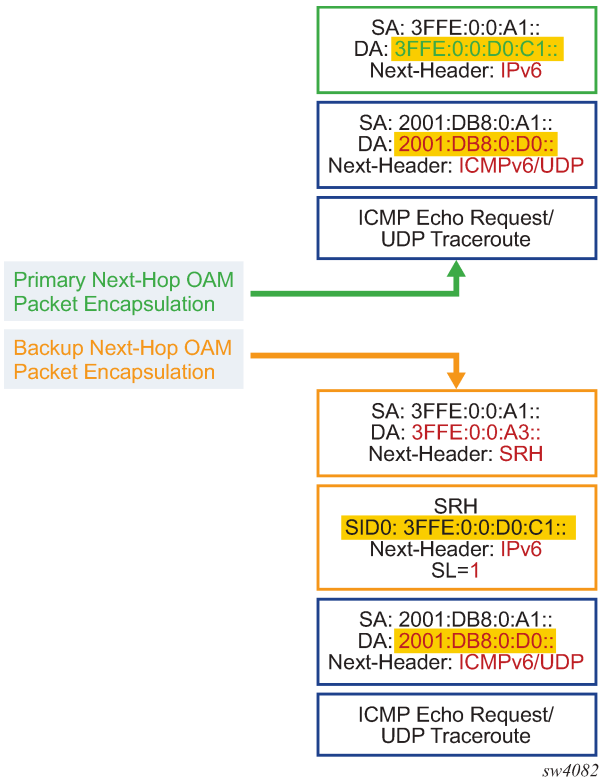
The CPM-originated ping or traceroute packet is encoded with:

- DA set to End.DT4, End.DT6, or End.DT46 SID
- the next-header field set to IPv6 (IPv6 VRF prefix)
- the outer IPv6 header hop-limit field is set to the default value 254
- the next-header field of the inner IPv6 header is set to ICMPv6 (for ping), to UDP or TCP (for traceroute)

The packet is encapsulated as shown in Figure 11: Packet encapsulation for ping or traceroute of a VRF IPv6 prefix over an SRv6 tunnel. The LFA SRH is also shown in the packet encapsulation of the SRv6 tunnel to node D when the TI-LFA or remote LFA repair tunnel is activated.

A similar encoding of the ping and traceroute packet is performed when testing a VRF IPv4 prefix that is resolved to an SRv6 tunnel. The difference is the inner packet is in an IPv4 packet format.

Figure 11: Packet encapsulation for ping or traceroute of a VRF IPv6 prefix over an SRv6 tunnel



4.7.2.2 Transit P router behavior

The packet is processed in the datapath, in the same manner as any SRv6 user-data packet, by the transit router.

4.7.2.3 Egress PE router behavior

At the target node, the packet is processed as follows:

- If the DA field of the outer IPv6 header matches a service SID and the payload type is IPv4 or IPv6, the datapath removes the SRv6 headers and extracts the inner IPv4 or IPv6 packet to the CPM.
- If the DA field of the packet matches a local locator prefix entry in the FIB and the payload type is either IPv4 or IPv6, the packet is handed to the SRv6 Forwarding Path Extension (FPE). The egress datapath of the SRv6 FPE removes the SRv6 headers and passes the inner IPv4 or IPv6 packet to the ingress datapath, which performs the regular exception handling for a ping or a traceroute packet.

4.7.3 Ping or traceroute of an IPv4 or IPv6 global routing instance prefix resolved to an SRv6 tunnel

This behavior is the same as described in Ping or traceroute of an IPv4 or IPv6 VRF prefix resolved to an SRv6 tunnel

5 OAM monitoring and reporting

OAM fault and performance tools monitor and report information about the network infrastructure and the services that rely on that infrastructure.

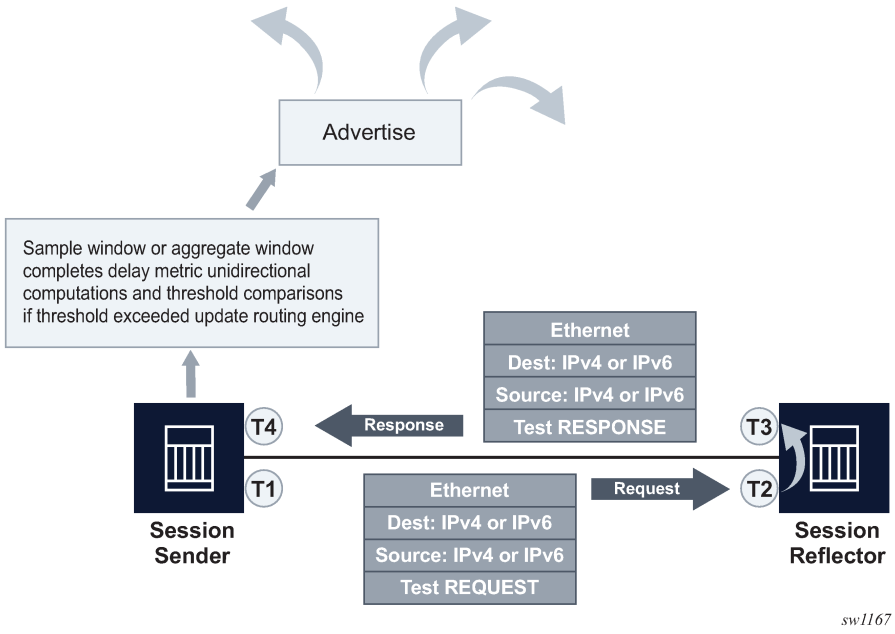
5.1 Link measurement

Note: This feature is supported on 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, 7250 IXR-X3b, 7730 SXR and 7250 IXR Gen 3 platforms.

Network elements use routing protocols to exchange information about local links, which can influence routing decisions. These interface attributes are typically static in nature. By using tools specifically designed to measure IP performance, dynamic unidirectional delay can be included in the advertised link attributes. A link measurement test is one such method for measuring and reporting delay information for directly connected IP peers. Link measurement uses the STAMP protocol defined in RFC 8762 to measure delay for an IP interface

The following figure shows directly connected IP interfaces and the link measurement interaction with routing.

Figure 12: Link measurement interactions



5.1.1 Link measurement template

Creation of a link measurement template is the first step of a link measurement test. The link measurement template is created by configuring the common test parameters using the **oam link-measurement** command. After the measurement template is created, an SR Linux interface references the link measurement template using the **oam link-measurement interface dynamic-measurement link-measurement-template** command. When the association between the interface and the template is established, the interface executes a process to determine the operational state of the test and detect any defect conditions that may prevent test execution. If no underlying conditions are present, the IP interface delay measurements are collected in measurement windows, compared with the configured thresholds, and reported to the routing engine for further processing when required.

Link measurement template parameters

The parameters that are configured using the **oam link-measurement interface dynamic-measurement link-measurement-template** command define the test criteria used by the test. Conceptually, the test criteria are divided into the following groups:

- General configuration
- Collection and reporting
- Protocol

Modifying a link measurement template

SR Linux supports the modification of active link measurement templates. That includes administratively disabling a link measurement template that IP interfaces are actively referencing. Modifying existing parameters causes interface delay tests that reference the modified template to terminate the current sample, and aggregate measurement windows, and start new measurement windows using the updated template parameters. The previous historical results are maintained, but the state field of the measurement window coinciding with the change indicates **Terminated**. Changing the description or the **last-reported-delay-hold** configuration does not cause a termination of the current sample and aggregate measurement windows.

Deleting a link measurement template

A link measurement template cannot be deleted if interfaces are referencing that template.

5.1.1.1 General configuration

The general configurations included in the **oam link-measurement measurement-template** command influence probe frequency, the delay metric type to monitor, and retention of the delay measurement last reported.

- The probe frequency, configured using the **interval** command, defines the transmission rate of the test packet.
- The **delay** command configures the delay metric (minimum, maximum, or average) that is used for comparison against any configured thresholds. This metric is the same for both types of measurement windows, the **sample-window** and **aggregate-sample-window**.
- The **unidirectional-measurement** command specifies the method used to compute the unidirectional delay. If the clock synchronization between nodal clocks used by the OAM timestamp function is

not synchronized to near exact accuracy, the **derived** option must be used. Specifying this option calculates the unidirectional measurement using the round trip delay divided by two computation. If synchronization can meet near exact accuracy, the **actual** option can be used. Specifying this option calculates the forward delay using the forward direction timestamps, T2-T1 computation.

- When the operational state of the link measurement test transitions to down, the OAM function instructs the routing engine to clear the last reported delay value at the expiration of the **last-reported-delay-hold** value. A previously reported delay is considered valid for the duration of this period and is cleared if the timer reaches zero. If the operational state returns to up before the timer expires, no action is taken to clear the previous value. The counter is reset to the configured value, waiting for the next operational down event. If the **last-reported-delay-timer** is set to zero, previously reported delay values from that test are cleared when the operational state changes to down without any additional time.

  An operational state is up for a test if:

  – the measurement template is administratively up

  – the source UDP port is available

  – an IPv4 or IPv6 protocol is administratively up

  – there are no internal errors that prevent the test from initializing

- The aging timer does not start a count to zero for failure conditions that do not affect the interface delay test operational state. The delay measurement last reported is maintained when conditions external to the interface delay test, such as fault conditions on the port, IP interface, routing changes, and so on, occur.

5.1.1.2 Collection and reporting

The collection and reporting parameters define the following:

- length of the **sample-window** and **aggregate-sample-window**. Two measurement windows are provided to support use cases that require a reporting hierarchy and both include the same configuration options.
- thresholds that trigger reporting. The threshold values determine when the measurement window updates the reported delay value.

Multiplier

The measurement windows use the **multiplier** command to determine the length of time that the measurement window remains open. The sample window length is multiples of the interval. This window stores the results of individual test probes for a total length of the **interval** multiplied by the **multiplier** value. The aggregate sample window multiplier length is the number of sample windows. This window stores the number of results passed from individual sample windows. In the aggregate sample window, the minimum, maximum, and average calculations are based on the results received from the sample window. For example, if the delay metric of interest is the average, the aggregate sample is a collection of averages passed from the sample window. The reporting in the aggregate sample window is as follows:

- The minimum is the minimum value for all the averages received.
- The maximum is the maximum value from all the averages received.
- The average is the average of all the averages received.

Window integrity

The comparison to thresholds and reporting decisions occurs at the end of the measurement window if it completes without termination and is deemed integral based on the **window-integrity** command configuration. Integrity is a percentage-based calculation that determines the number of samples that must be present in the measurement window for that window to be considered integral. If the number of samples in the window equals or exceeds the number of required samples, the result is treated as representative and follows normal post-measurement window processing. However, if the number of samples in the window does not achieve integrity, the result is not considered representative and is only recorded for historical purposes, but is otherwise ignored and not processed. By default, integrity checking is disabled and all results from a measurement window are treated as integral and compared to the configured thresholds.

Threshold

There are two types of thresholds:

- a microsecond increase or decrease, configured using the **absolute** command
- a percentage increase or decrease, configured using the **relative** command

Thresholding compares the measurement window result to the delay measurement last reported at the end of the successful (completed) measurement window. Reporting is on a per-threshold, per-measurement window basis. If multiple thresholds are reached for a completed measurement window, only one threshold triggers an update to the routing engine.

Configuration of the measurement windows depends on the specific solution requirements. The two measurement windows collect information regardless of configured thresholds. Both types of measurement windows support their own threshold and integrity configuration. By default, thresholds for both measurement windows are disabled; that is, neither window can report any values to the routing engine.

Reporting

Reporting is enabled by default and the routing engine is informed of threshold events. The notification process uses the **reporting** command and threshold configurations. At least one threshold must be configured to report to the routing engine. Disabling reporting allows the function to execute but prevents the reporting of threshold events to the routing engine. If this value is toggled and a value was previous reported, the reported value is cleared, and the process returns to the initial reporting phase.

5.1.1.3 Protocol

The **oam link-measurement measurement-template stamp** command parameters influence the format of the test packet, processing, QoS handling, IPv6 discovery, and return path.

Source UDP port

By default, link measurement uses dynamic source UDP ports. However, a specific source UDP port can be configured if required from the range of source UDP ports allocated to STAMP is 64374 to 64383 if required. The UDP port must previously be allocated to the link-measurement application.

```
--{ + candidate shared default }--[  ]--
# info oam ippm
    oam {
        ippm {
```

```
            source-udp-port-pools {
                port 64374 {
                    application-assignment oam-pm-ip
                }
            }
        }
    }
```

IPv6 destination discovery

IPv6 destination address discovery allows the discovery of a single directly connected IPv6 peer. When this option is enabled, a bootstrap function using an ICMPv6 echo request with a destination ff02::2 is generated. When the directly-connected peer responds, the link measurement function uses the source address of the ICMPv6 echo response as the destination address for the link measurement test packets.

The process has four main components:

- Enabling the functions using **admin-state** command.
- Configuring the **discovery-interval** command. This is the initial timer used by the discovery process to discover the peer. This interval is used for the duration of the **discovery-timer**.
- Implementing the discovery phase. If the timer expires or the peer is discovered before the expiration of the **discovery-timer**, the process reverts back to the **update-interval**.
- Implementing the **update-interval**. This is an optional maintenance component of the peer address that runs at a slower rate. This option is not required and can be disabled in environments where the peer address is unlikely to change. If the peer is not discovered during the **discovery-timer** and with the **update-interval** disabled, the peer fails to be discovered. Disable and then enable the IPv6 protocol to restart the discovery process.

Return path

By default, the session reflectors use routing to return the response packet to the session sender. There are instances when it may be beneficial to be selective about the IP interface used for the return path. For example, when multiple tests are executed on different interfaces between the same pair of nodes, and using non-directly connected interface addresses, and ECMP exists between the two nodes. In this case, the **return-path link** command can be configured as **true**. This includes the return path TLV and link sub TLV in the test packet. This configuration instructs the session reflector to send the response out to the same IP interface on which it was received. The destination IP address for the response packet must be installed in the forwarding table and reachable from that interface. If the routing engine determines that the prefix is not reachable from that interface, the response packet is dropped at the reflector.

5.1.2 Interface assignment

The test criteria-specific link measurement is configured in the link measurement template. The delay test is executed from the network instance with the type default and requires and interface that is part of the network instance **oam link-measurement interface** command. The link measurement template does not include interface-specific requirements, such as the IP protocol encapsulating the test packet or IP source and destination addressing.

5.1.2.1 IP addressing

To enable dynamic measurements for the interface, configure a link measurement template and enable the test protocol using the **oam link-measurement interface dynamic-measurement protocol ipv4** or **oam link-measurement interface dynamic-measurement protocol ipv6** command. Only one protocol, IPv4 or IPv6, can be enabled for an interface delay test at any time. Interfaces defined as **loopback** do not support interface delay tests and are an invalid interface type.

IPv4 address auto-discovery

When the IPv4 protocol is enabled with no addressing configured, the source address is automatically assigned to the primary IPv4 address of the IP interface. The destination address is automatically assigned if the primary IPv4 address has a prefix length of 30 or 31. In other cases, such as shorter prefix lengths or unnumbered interfaces, the destination address cannot be resolved and must be configured manually. The **source-ip** and **destination-ip** commands take precedence over the auto-assigned addressing; the IPv4 addresses must be unicast.

IPv4 auto-assigned addressing is not updated for operationally up interface delay tests when the IP addressing associated with that interface is changed. Nokia suggests the following options to update the auto-assigned addressing:

- administratively disable and enable the protocol used for the interface delay test
- disable and enable the IP interface under which the IP address has changed

IPv6 address auto-discovery

When the IPv6 protocol is enabled without any source address, the system uses the link-local address associated with the interface as the source. If there is no destination address configured, the destination discovery process is initiated if the associated link measurement template assigned to this interface has the following command enabled.

```
oam link-measurement measurement-template ip ipv6-destination-discovery
```

5.1.2.2 Test initialization

When the link measurement template is assigned to an IP interface, the audit process determines the operational state of the test. The interface delay test transitions to operationally up if the following conditions are met:

- the associated measurement template is administratively enabled
- there is an administratively enabled test protocol configured using the **ipv4** or **ipv6** command
- the system resources are available to start the test

Further validation determines if there are any underlying conditions that are considered detectable transmission errors, which are listed in the following table.

When all audit conditions successfully pass, the test begins. When no thresholds are configured, the test collects delay information as history, but without at least one configured threshold value, reporting updates to the routing engine are disabled. If at least one threshold is configured, the interface enters the first report phase. Because no previous delay value has been reported, the first measurement window

with a configured threshold that completes with integrity triggers the delay measurement report. After this benchmark is set, all subsequent thresholds use the delay measurement last reported as the comparison.

5.1.2.3 History and results

Active interface delay tests retain 50 sample windows and 20 aggregate sample windows in history. The current measurement windows and historical results are not maintained across CPM switchovers. The delay measurement last reported is maintained after a CPM switchover to retain the baseline. The interface does not enter the first reporting phase following a CPM switchover.

The results can be viewed using the **info from state oam link-measurement interface** command.

5.1.3 Allocating source UDP port to link measurement

Procedure

To allocate a source UDP port to link measurement, you specify the UDP port number in the **oam ippm source-udp-port-pools port** command and select **link-measurement** option in the **application-assignment** parameter.

Example: Allocating source USP port to link measurement

This example allocates a source UDP port to perform the link measurement test.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ippm
    oam {
        ippm {
            source-udp-port-pools {
                port 64374 {
                    application-assignment link-measurement
                }
            }
        }
    }
```

5.1.4 Performing link measurement test

About this task

Perform the following steps to carry out the link measurement test:

Procedure

Step 1. Perform the following steps to create a link measurement template:

a. Use the **oam link-measurement measurement-template** command to create a measurement template.

b. Enable the measurement template and configure the parameters as shown in Creating a measurement template.

Step 2. Perform the following steps to assign an interface to the link measurement template:

a. Use the **oam link-measurement interface** command to create an interface.

b. Use the **oam link-measurement interface interface-ref** command to provide a reference to an interface and subinterface.

c. Use the **oam link-measurement interface dynamic-measurement link-measurement link-measurement-template** command to assign the link measurement template.

d. Use the **oam link-measurement interface dynamic-measurement protocol** command to configure the protocol and IP address of the source and destination, as shown in Assigning an interface to the link measurement template.

Step 3. When the link measurement template is assigned to an IP interface, the audit process determines the operational state of the test. Further validation determines if there are any underlying conditions that are considered detectable transmission errors. When all audit conditions successfully pass, the delay collection begins.

Step 4. Use the following commands to view the results of the link measurement:

- **info from state with-context oam link-measurement interface** - See Displaying link measurement test results.
- **info from state with-context oam link-measurement resources**. See Displaying link measurement resources.
- **info from state oam link-measurement interface * | as table** - See Displaying link measurement interface operation state table
- **info from state oam link-measurement interface * statistics aggregate-sample-window index * | as table** - See Displaying link measurement interface statistics aggregate sample window table
- **info from state oam link-measurement interface * statistics sample-window index * | as table** - See Displaying link measurement interface statistics sample window table

Example: Creating a measurement template

This example creates a measurement template.

```
--{ + candidate shared default }--[  ]--
# info with-context oam link-measurement measurement-template link-1
    oam {
        link-measurement {
            measurement-template link-1 {
                admin-state enable
                sample-window {
                    multiplier 30
                    window-integrity 70
                    threshold {
                        absolute 50
                    }
                }
                ip {
                    destination-udp-port 862
                    ttl 1
                }
                pdu-type {
                    stamp {
                    }
                }
            }
        }
```

```
    }
```

Example: Assigning an interface to the link measurement template

This example assigns an interface to the link measurement template.

```
--{ + candidate shared default }--[  ]--
# info with-context oam link-measurement interface lm01
    oam {
        link-measurement {
            interface lm01 {
                interface-ref {
                    interface ethernet-1/1
                    subinterface 1
                }
                dynamic-measurement {
                    link-measurement {
                        link-measurement-template link-1
                    }
                    protocol {
                        ipv4 {
                            admin-state enable
                            destination-ip 192.168.1.2
                            source-ip 192.168.1.1
                        }
                    }
                }
            }
        }
    }
```

Example: Displaying link measurement test results

This example displays the results of the link measurement test.

```
--{ + candidate shared default }--[  ]--
A:admin@sxr1a# info from state with-context oam link-measurement interface lm01
    oam {
        link-measurement {
            interface lm01 {
                oper-state up
                detectable-transmit-error none
                operational-source-address 192.168.1.1
                source-ip-auto-assigned false
                operational-destination-address 192.168.1.2
                destination-ip-auto-assigned false
                in-use-source-udp-port 54000
                in-use-destination-udp-port 862
                stamp-session-identifier 10
                reporting true
                last-reported-dynamic-delay 50
                report-timestamp 2026-01-28T17:54:01.000Z
                report-triggered-by sample-threshold-absolute
                aggregate-newest-index 2
                sample-newest-index 2
                interface-ref {
                    interface ethernet-1/1
                    subinterface 1
                }
                dynamic-measurement {
                    link-measurement {
                        link-measurement-template link-1
                    }
```

```
                    protocol {
                        ipv4 {
                            admin-state enable
                            destination-ip 192.168.1.2
                            source-ip 192.168.1.1
                        }
                        ipv6 {
                            admin-state disable
                        }
                    }
                }
                statistics {
                    aggregate-sample-window {
                        index 1 {
                            end-timestamp-utc 2026-01-28T17:54:01.000Z
                            window-state sw-reported
                            sample-window-count 1
                            minimum 0
                            maximum 0
                            average 0
                            result 0
                            integrity false
                        }
                        index 2 {
                            end-timestamp-utc 1970-01-01T00:00:00.000Z
                            window-state in-progress
                            sample-window-count 0
                            minimum 0
                            maximum 0
                            average 0
                            result 0
                            integrity false
                        }
                    }
                    sample-window {
                        index 1 {
                            end-timestamp-utc 2026-01-28T17:54:01.000Z
                            window-state sw-reported
                            transmitted-packets 30
                            received-packets 30
                            minimum 50
                            maximum 68
                            average 56
                            result 50
                            integrity true
                            error-count 0
                            stamp-unrecognized-flag-count 0
                            stamp-malformed-flag-count 0
                            zero-or-negative-delay-count 0
                            duplicate-packet-count 0
                        }
                        index 2 {
                            end-timestamp-utc 1970-01-01T00:00:00.000Z
                            window-state in-progress
                            transmitted-packets 1
                            received-packets 1
                            minimum 0
                            maximum 0
                            average 0
                            result 0
                            integrity false
                            error-count 0
                            stamp-unrecognized-flag-count 0
                            stamp-malformed-flag-count 0
```

```
                            zero-or-negative-delay-count 0
                            duplicate-packet-count 0
                        }
                    }
                }
            }
        }
    }

    link-measurement {
        measurement-template link-1 {
            admin-state enable
            sample-window {
                multiplier 30
                window-integrity 70
                threshold {
                    absolute 50
                }
            }
            ip {
                destination-udp-port 862
                ttl 1
            }
            pdu-type {
                stamp {
                }
            }
        }
        interface lm01 {
            interface-ref {
                interface ethernet-1/1
                subinterface 1
            }
            dynamic-measurement {
                link-measurement {
                    link-measurement-template link-1
                }
                protocol {
                    ipv4 {
                        admin-state enable
                        destination-ip 192.168.1.2
                        source-ip 192.168.1.1
                    }
                }
            }
        }
    }
```

Example: Displaying link measurement resources

```
--{ + candidate shared default }--[  ]--
# info with-context from state oam link-measurement resources
    oam {
        link-measurement {
            resources {
                template {
                    in-use 1
                }
                interface {
                    in-use 1
                }
            }
        }
```

```
    }
```

Example: Displaying link measurement interface operation state table

This example shows the link measurement interface operation state table.

```
--{ + candidate shared default }--[  ]--
# info from state oam link-measurement interface * | as table
+-----+------+----------+----------+----------+-------------+----------+----------+-----------+
|Name |Oper- |Detectable|Stamp-    |Reporting |Last-reported| Report-  | Sample-  |Operational|
|     |state |transmit  |session-  |          |dynamic      | timestamp| newest-  |  failure  |
|     |      |error     |identifier|          |delay        |          |  index   |           |
+=====+======+==========+==========+==========+=============+==========+==========+===========+
|lm001| down | subinter |    0     | false    |    none     | 1970-01- |    0     | template  |
|     |      | face-    |          |          |             | 01T00:00 |          | -admin-   |
|     |      | down     |          |          |             | :00.000Z |          | down      |
|lm01 | up   | subinter |  1000    | false    |    none     | 1970-01- |   1249   |           |
|     |      | face-    |          |          |             | 01T00:00 |          |           |
|     |      | down     |          |          |             | :00.000Z |          |           |
+-----+------+----------+----------+----------+-------------+----------+----------+-----------+
```

Example: Displaying link measurement interface statistics aggregate sample window table

This example shows the link measurement interface statistics aggregate sample window table.

```
--{ + candidate shared default }--[  ]--
# info from state oam link-measurement interface * statistics aggregate-sample-window index * |
as table
+---------+---------+----------+---------+---------+----+----+---------+---------+---------+
|Interface|  Index  | End-time | Window- | Sample- | Min| Max| Average | Result  |Integrity|
|         |         | stamp-   |  state  | window- |    |    |         |         |         |
|         |         |   utc    |         |  count  |    |    |         |         |         |
+=========+=========+==========+=========+=========+====+====+=========+=========+=========+
| lm01    |      89 |2025-01-  |completed|    0    |  0 |  0 |    0    |    0    | true    |
|         |         |27T19:4   |         |         |    |    |         |         |         |
|         |         |8:19.00 0Z|         |         |    |    |         |         |         |
| lm01    |      90 |2025-01-  |completed|    0    |  0 |  0 |    0    |    0    | true    |
|         |         |27T19:5   |         |         |    |    |         |         |         |
|         |         |0:19.00 0Z|         |         |    |    |         |         |         |
+---------+---------+----------+---------+---------+----+----+---------+---------+---------+
```

Example: Displaying link measurement interface statistics sample window table

This example shows the link measurement interface statistics sample window table.

```
--{ + candidate shared default }--[  ]--
# info from state oam link-measurement interface * statistics sample-window index * | as table
+-------+-------+---------+---------+--------+-------+----+----+-------+-------+-------+-------+
| Inter | Index |End-time |Window   |Trans-  |Recei- | Min| Max|Average|Result | Integ | Error |
| face  |       |stamp-utc|state    |mitted- |ved-   |    |    |       |       | rity  |   -   |
|       |       |         |         |packets |packets|    |    |       |       |       | count |
+=======+=======+=========+=========+========+=======+====+====+=======+=======+=======+=======+
| lm01  |  1333 |2025-01  |completed|     0  |     0 |  0 |  0 |     0 |       | true  |     0 |
|       |       |27T20    |         |        |       |    |    |       |       |       |       |
|       |       |:32:2    |         |        |       |    |    |       |       |       |       |
|       |       |9.000 z  |         |        |       |    |    |       |       |       |       |
| lm01  |  1334 |2025-01  |completed|     0  |     0 |  0 |  0 |     0 |       | true  |     0 |
|       |       |27T20    |         |        |       |    |    |       |       |       |       |
|       |       |:32:3    |         |        |       |    |    |       |       |       |       |
|       |       |9.000 z  |         |        |       |    |    |       |       |       |       |
| lm01  |  1335 |2025-01  |completed|     0  |     0 |  0 |  0 |     0 |       | true  |     0 |
|       |       |27T20    |         |        |       |    |    |       |       |       |       |
```

```
|       |       |:32:4    |         |        |       |    |    |       |       |       |       |
|       |       |9.000 z  |         |        |       |    |    |       |       |       |       |
+-------+-------+---------+---------+--------+-------+----+----+-------+-------+-------+-------+
```

5.2 Performance monitoring

Note: This feature is supported on 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, 7250 IXR-X3b, 7730 SXR and 7250 IXR Gen 3 platforms.

Performance monitoring encompasses a variety of tools and protocols designed to measure, report, analyze, and optimize network performance, allowing network administrators to detect and resolve issues proactively. This chapter provides information about delay and loss measurement as part of STAMP OAM performance monitoring.

5.2.1 STAMP OAM performance monitoring

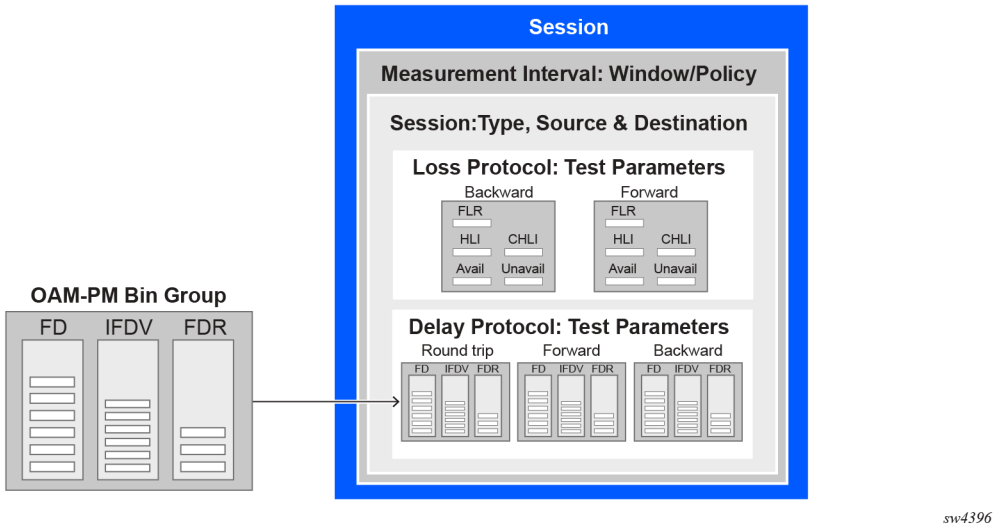
Note: This feature is supported on 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, 7250 IXR-X3b, and 7730 SXR platforms.

The STAMP OAM performance monitoring architecture for gathering and computing Key Performance Indicators (KPIs) using standard protocols and a robust collection model consists of the following foundational components:

- session: This is the overall collection of the test parameters, measurement intervals, thresholds and storage of results. It is the overall container that defines the attributes of the session.
- standard performance monitoring packets: SR Linux supports STAMP to measure performance metrics such as delay and packet loss. See Session sender packet format and Session reflector packet format in the STAMP chapter for more information about STAMP test packets.
- measurement interval: These are time-based non-overlapping windows that capture results that are received in that window of time.
- data structures: These are the unique counters and measurement results that represent the specific protocol.
- bin group: These are ranges in microseconds that count the results that fit into the range.

The following figure shows the hierarchy of the architecture. This figure intends to show the relationship between the components and not to depict all the details of the required parameters.

Figure 13: OAM performance monitoring architecture hierarchy



5.2.1.1 Session

The session is the overall collection of test information fields. The session can be viewed as the single container that combines all aspects of individual tests and the various OAM-PM components. The **session** command includes parameters such as:

- **session-type**: The session type is either proactive or on-demand. The session type setting influences the individual test timing parameters.
- **test-id**: The test identifier is a configured numerical value or using the **auto** keyword. An automatically assigned test-id is released when the test is deleted. Any action that causes the test to be deleted and recreated releases the original test identifier and allocates a new one. These test identifiers are not persistent and not maintained across CPM switchovers. New test identifiers are allocated in the case of CPM switchover.
- test parameters: These include start time, stop time, ability to activate a test, and ability to deactivate a test.
- **measurement-interval**: This is the assignment of collection windows to the session with the appropriate configuration parameters .

Session types and operational states

The operational state of a session is influenced by:

- session type
- session administrative state

In a proactive session, the operational state is up when the administrative state is enabled. The operational state is down when the administrative state is disabled. A proactive test session starts immediately after the **oam performance-monitoring ip session stamp admin-state** command is set to **enable**. A proactive test session stops immediately after the **oam performance-monitoring ip session stamp admin-state**

command is set to **disable**. The operational state of a proactive session mirrors the administrative state of the session.

In an on-demand session, the operational state is up when the administrative state is enabled, and the **tools oam performance-monitoring ip session on-demand-action start** command is executed. The on-demand test stops after one of the following actions:

- the completion of the **test-duration** configured using the **oam performance-monitoring ip session session-type on-demand stamp test-duration** command
- the execution of the **tools oam performance-monitoring ip session on-demand-action stop** command

The operational state of an on-demand sessions is not directly tied to the administrative state. It depends on the initiation and completion of tests.

5.2.1.2 Standard PM packets

SR Linux supports STAMP (Simple Two-way Active Measurement Protocol) to measure performance metrics such as delay and packet loss. STAMP defines test packets in two directions:

- session sender packet: test request packets that the session sender transmits to the session reflector.
- session reflector packet: test response packets that the session reflector sends to the session sender.

Refer to the **Session sender packet format** and **Session reflector packet format** in the **STAMP** chapter for detailed information about STAMP test packets.

5.2.1.3 Data structures

There are two main metrics that are the focus of OAM performance monitoring: delay and loss.

Delay metrics

SR Linux supports the following delay metrics:

- Frame Delay (FD): This measures the time taken for a packet to traverse the network. Any negative FD values are set to zero for binning purposes.
- Frame Delay Range (FDR): This represents the difference between the FD and the lowest FD recorded within a measurement interval. For the first interval, the minimum delay is set to zero. For subsequent intervals, the minimum delay of the previous measurement interval is used as the reference. Negative FD values are set to zero before calculating FDR.
- Inter-Frame Delay Variation (IFDV): Also known as jitter, IFDV measures the variation in delay between adjacent test frames. The absolute difference between the current and previous delay values is calculated, even if the previous delay was negative.

FD, FDR, and IFDV are categorized into bins based on the bin-group configuration. The minimum, maximum, and average values for each direction (forward, backward, and round-trip) are also reported. Delay threshold events and the last time the threshold crossing alarm (TCA) was triggered are also logged .

By default, the average for all delay metrics includes all the results within the measurement interval. However, it is possible to exclude the measurements using **exclude-from-average** for a specified

direction. The results are binned but the delay values included in the exclude option are not included in the average computation.

Loss metrics

SR Linux supports the following loss metrics:

- out loss: This metric measures the difference between the packets that the session reflector receives and those that the session sender transmits.
- in loss: This metric calculates the difference between the packets that the session sender receives and those that the session reflector transmits.
- Frame Loss Ratio (FLR): This percentage metric represents the ratio of lost packets during times of availability. FLR is not incremented during periods of unavailability.
- available: The number of delta-ts that are recorded as available. These delta-ts do not exceed the FLR threshold and do not follow an unavailable state. If the available delta-ts follow an unavailable state, they need to fill the availability window before transitioning to available.
- unavailable: The number of delta-ts that are recorded as unavailable. These delta-ts exceed the FLR threshold and do not follow an available state. If the unavailable delta-ts follow and available state, they need to fill the unavailability window before transitioning to unavailable.
- undetermined availability: This counter increments when packets are lost and there is no explicit information about the fate of the packet. This occurs after timeouts and follows an available state.
- undetermined unavailability: Similar to undetermined availability, this counter increments after packet timeouts following an available state when there is no explicit information about the fate of the packet.
- High Loss Interval (HLI): This increments when individual delta-ts reach or exceed the FLR configuration. By default, this is calculated during the availability periods. Executing the **oam performance-monitoring ip session stamp loss hli-force-count** command and configuring the **true** option increments the HLI regardless of the availability state.
- Consecutive High Loss Interval (CHLI): This increments when consecutive HLI intervals meet or exceed a specified threshold within the sliding window. CHLI increments only a single time for each availability window.

5.2.1.4 Measurement intervals

A measurement interval is a window of time that compartmentalizes the gathered measurements for an individual test that has occurred during that time. Allocation of measurement intervals, which equates to system memory, is based on the metrics being collected. This means that when both delay and loss metrics are collected, they allocate their own set of measurement intervals.

Duration

The measurement interval durations are as follows:

- **1-min**
- **5-min**
- **15-min**
- **1-hour**
- **1-day**

Boundary type

The **boundary-type** parameter defines the start of the measurement interval and can be aligned to the local time-of-day clock, with or without an optional offset. By default, the start boundary is **clock-aligned** without an offset. When this configuration is deployed, the measurement interval establishes non-overlapping time-based windows which complete at the specified time. The **boundary-type** parameter can be aligned using the **test-aligned** option, which means that the start of the measurement interval coincides with the activation of the test and the length of the measured interval determines the completion.

Clock aligned

When a boundary is **clock-aligned** and **clock-offset** option is configured, a specified amount of time is applied to the measurement interval. Offsets are configured on a per-measurement interval basis and only applicable to clock-aligned measurement intervals. Only offsets less than the measurement interval duration are allowed. The following table lists examples of the start times of each measurement interval.

Table 12: Measurement interval start times

Offset	1-min	15-min	1-hour	1-day
0 (default)	0, 1, 2, 3, ...	00, 15, 30, 45	00 (top of the hour)	midnight
10 minutes	rejected	10, 25, 40, 55	10 min after the hour	10 min after midnight
30 minutes	rejected	rejected	30 min after the hour	30 min after midnight
60 minutes	rejected	rejected	rejected	01:00 AM

Test aligned

Although **test-aligned** approaches may seem beneficial for simplicity, there are some drawbacks that need to be considered. The goal of the time-based and well-defined collection windows allows for the comparison of measurements across common windows of time throughout the network and for relating different tests or sessions. On-demand tests are typically used for troubleshooting or short term monitoring that does not require alignment or comparison to other PM data and may make better use of the test-aligned boundary.

Intervals stored

The statistical data collected and the computed results from each measurement interval are maintained in volatile system memory . The number of intervals stored is configurable per measurement interval. Different measurement intervals have different defaults and ranges. The **interval-stored** parameter defines the number of completed individual test runs to store in volatile memory. There is an additional allocation to account for the active measurement interval.

If the retained test data for a measurement interval consumes the final entry, any subsequent entries cause the removal of the oldest data.

Threshold events

The following are the two types of threshold events:

- stateless. The stateless threshold events are:

  - autonomous. Each measurement interval operates independently without carrying forward any information about events from previous intervals.
  - self-contained. The events are evaluated and triggered within the confines of a single measurement interval.
  - enacted when **clear-threshold** is unset.
- stateful. The stateful threshold events are:
  - persistent. The events remain active until specific clear-threshold conditions are met at the end of a subsequent interval.
  -
  - enacted when the clear-threshold is set within the configured range. A value of zero indicates the event clears if no results fall within the specified range in a subsequent interval.

Delay event

Counter-based events. These are simple counts that compare to the raise-threshold for raising and the clear-threshold for clearing. These are per direction, forward, backward and round-trip. Each of these delay thresholds are raised a maximum of one in a measurement interval when the count in the specified bins reach the raise-threshold. The types of delay events are as follows:

- Frame Delay (FD)
- Frame Delay Range (FDR)
- Inter-Frame Delay Variation (IFDV)

Each of these delay threshold events are raised a single time in a measurement interval immediately after the threshold is reached.

Loss events

The types of loss events are as follows:

- Counter-based events. These are simple counts that compare to the **raise-threshold** for raising and the **clear-threhsold** for clearing. The standard directions, forward and backward as well as a mathematical aggregate that is computed by summing the forward and backward values, are supported. Each of these loss threshold events are raised a maximum of one time in a measurement interval when the count in the specified bins reach the raise-threshold.
  - High Loss Interval (HLI) event
  - Consecutive High Loss Interval (CHLI) event
  - unavailability event
  - undetermined availability event
  - undetermined unavailability event
- Average Frame Loss Ratio (Avg-FLR) event. Unlike other loss events, the Avg-FLR event is raised only at the end of the measurement interval. This event does not support aggregated computation. It supports forward and backward directions.

Loss event template

The loss event template is created using the **oam performance-monitoring ip loss loss-events-template** command. All of the loss events are configured using this template. During loss measurement, the loss event template is referenced by a performance monitoring session.

The following considerations apply to loss event templates:

- A loss event template cannot be deleted if it is referenced by a performance monitoring session.
- A loss event template can be modified even if it is referenced by a performance monitoring session without disrupting the ongoing session.
- The reference changes that include, adding a new reference or deleting an existing reference within a loss test session can be done without impacting the performance monitoring session or the loss test session.
- The configuration changes made to loss event templates affect only the loss event function ensuring that performance monitoring continues seamlessly.
- The operational states of the loss events transition based on the configuration changes and the timing of measurement intervals.

5.2.1.5 Bin group

A bin group is a collection of bins where each bin represents a range of delay values. When a delay measurement is performed, the delay results are stored in appropriate bins based on its value. This process is known as binning and it allows for a structured and aggregated view of delay performance over time. Bin groups are created using the **oam performance-monitoring ip delay bin-group** command and referenced by the session using the **oam performance-monitoring ip session stamp delay bin-group** command.

Bin type

There are three types of binnable delay metrics:

- frame delay (FD)
- inter-frame delay variation (IFDV)
- frame delay range (FDR)

All bin types are available in the forward, backward, and round-trip directions. Each of these metrics can have up to ten bin groups configured to group the results.

Bin boundary

Bin groups are configured by indicating a lower boundary. Bin 0 has a lower boundary that is always zero and is not configurable. The microsecond range of the bins is the difference between the adjacent lower boundaries.

For example, **bin-type fd bin 1** configured with **lower-bound 1000** means that:

- bin 0 captures all frame delay statistics results between 0 and 1 ms
- bin 1 captures all results above 1 ms and below the bin 2 lower boundary, bin 2 is not shown

The last bin configured represents the bin that collects all the results at and above that lower-bound value. Not all ten bins have to be configured.

Each delay type requires their own values for the bin groups. It is not possible to configure a bin with different values for round-trip, forward, and backward. Consider the configuration of the boundaries that represent the important statistics for that specific requirement.

Bin group 1 is the default bin group. Every session requires a bin group to be assigned. By default, bin group 1 is assigned to every performance monitoring session that does not have a bin group explicitly configured. Bin group 1 cannot be modified. Bin group 1 is an automatically crated object and not visible in the configuration. If the bin-group 1 is added to the configuration only the mandatory default configuration values may be added. It bin-group 1 is added to the configuration its behavior will non default bin-groups.

Bin group behaviour

The following considerations apply for bin groups:

- bin groups cannot be deleted if referenced by a performance monitoring session
- bin groups cannot be disabled if referenced by a delay test with the **admin-state** parameter set to *enable*
- bin groups can be modified even if referenced by a delay test regardless of the **admin-state** parameter setting. Delay results for the performance monitoring session referencing a changed bin-group will be deleted and a new set of bins will start recording results.
- bin group that is excluded from average can be modified even if referenced by a delay test with the **admin-state** parameter set to *enable*
- any changes to the attributes of the **oam performance-monitoring ip delay bin-group bin-type delay-event** command do not affect the performance monitoring session or test sessions

5.2.2 Configuring an IP OAM-PM session

This section provides information about how to configure an IP OAM-PM session and examples of common configurations.

5.2.2.1 Configuring a STAMP OAM-PM session

Procedure

To configure a STAMP OAM performance monitoring session, use the **oam performance-monitoring ip session** command and configure the parameters as shown in Configuring a STAMP OAM performance monitoring session.

A range of source UDP ports is allocated to STAMP. These must be assigned to the appropriate STAMP application before they can be configured under the application.

To allocate a source UDP port to the OAM STAMP application, use the **oam ippm source-udp-port-pools port 64374 application-assignment oam-pm-ip** command and configure the parameters as shown in the example, Allocating source UDP port to OAM PM application. Configuration of the source UDP port should only be used when explicitly required. If not configured, the source UDP port is dynamically allocated from the dynamic UDP port range by the OAM performance monitoring application.

Example: Configuring a STAMP OAM performance monitoring session

This example configures a STAMP OAM performance monitoring session.

```
--{ + candidate shared default }--[  ]--
# info with-context oam performance-monitoring ip session session1
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    description test
                    session-type proactive
                    destination-ip 10.2.1.2
                    destination-udp-port 862
                    source-ip 10.2.1.1
                    network-instance default
                    dscp CS6
                    profile in
                    ttl 255
                    measurement-interval 1-minute {
                        clock-offset 0
                        intervals-stored 32
                        threshold-alerts {
                            loss-event disable
                            delay-event disable
                        }
                    }
                    stamp {
                        admin-state enable
                        interval 1s
                        delay {
                            bin-group gp1
                        }
                        loss {
                            flr-threshold 10
                            hli-force-count true
                            timing {
                                frames-per-delta-t 10
                                consecutive-delta-t 5
                                chli-threshold 2
                            }
                        }
                    }
                }
            }
        }
    }
```

Example: Allocating source UDP port to OAM PM application

This example allocates a source UDP port to the OAM PM application.

```
--{ + candidate shared default }--[  ]--
# info with-context oam ippm source-udp-port-pools
    oam {
        ippm {
            source-udp-port-pools {
                port 64374 {
                    application-assignment oam-pm-ip
                }
            }
        }
```

```
    }
```

5.2.2.2 Performing STAMP OAM-PM delay measurement

About this task

Perform the following steps to measure STAMP OAM-PM packet delay:

Procedure

Configure a bin group

Step 1. To configure a bin group, use the **oam performance-monitoring ip delay bin-group** command and specify the parameters as shown in the following example.

Example

Configuring a bin group

```
--{ + candidate shared default }--[  ]--
# info with-context oam performance-monitoring ip delay bin-group gp1
    oam {
        performance-monitoring {
            ip {
                delay {
                    bin-group gp1 {
                        admin-state enable
                    }
                }
            }
        }
    }
```

Configure the bin type.

Step 2. To configure the bin type, use the **oam performance-monitoring ip delay bin-group bin-type** command. Specify the delay event and exclude from average parameters.

Example

Configuring a bin type

```
--{ + candidate shared default }--[  ]--
# info with-context oam performance-monitoring ip delay bin-group gp1
    oam {
        performance-monitoring {
            ip {
                delay {
                    bin-group gp1 {
                        admin-state enable
                        bin-type fd {
                            bin 1 {
                                lower-bound 1000
                            }
                            bin 2 {
                                lower-bound 2000
                            }
                            bin 3 {
                                lower-bound 3000
                            }
                            bin 4 {
```

```
                            lower-bound 5000
                        }
                        bin 5 {
                            lower-bound 7000
                        }
                        bin 6 {
                            lower-bound 10000
                        }
                        bin 7 {
                            lower-bound 13000
                        }
                        bin 8 {
                            lower-bound 16000
                        }
                        bin 9 {
                            lower-bound 20000
                        }
                    }
                    bin-type ifdv {
                        bin 1 {
                            lower-bound 100
                        }
                        bin 2 {
                            lower-bound 200
                        }
                        bin 3 {
                            lower-bound 300
                        }
                        bin 4 {
                            lower-bound 400
                        }
                        bin 5 {
                            lower-bound 500
                        }
                        bin 6 {
                            lower-bound 600
                        }
                        bin 7 {
                            lower-bound 700
                        }
                        bin 8 {
                            lower-bound 800
                        }
                        bin 9 {
                            lower-bound 900
                        }
                    }
                }
            }
        }
    }
```

Configure the session type.

Step 3. Perform one of the following:

a. To configure a proactive test session, use the **oam performance-monitoring ip session session-type proactive** command.

Example

Configuring a proactive test session

```
--{ + candidate shared default }--[  ]--
# info with-context oam performance-monitoring ip session session1 session-type
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    session-type proactive
                }
            }
        }
    }
```

b. To configure an on-demand test session, use the **oam performance-monitoring ip session session-type on-demand** command.

Example

Configuring an on-demand test session

```
--{ +* candidate shared default }--[  ]--
# info with-context oam performance-monitoring ip session session1 session-type
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    session-type on-demand
                }
            }
        }
    }
```

Enable STAMP and configure delay measurement parameters.

Step 4. To enable STAMP and configure the delay measurement parameters, use the **info oam performance-monitoring ip session stamp** command and specify the parameters as shown in the example.

Example

Configuring STAMP parameters for delay measurement

```
--{ + candidate shared default }--[  ]--
A:srl1# info with-context oam performance-monitoring ip session session1
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    destination-ip 192.0.2.1
                    destination-udp-port 862
                    source-ip 192.0.2.2
                    source-udp-port 64374
                    network-instance default
                    measurement-interval 1-minute {
                        boundary-type clock-aligned
                        clock-offset 0
                    }
                    stamp {
                        admin-state enable
                        test-id auto
```

```
                        interval 1s
                        delay {
                            bin-group gp1
                        }
                    }
                }
            }
        }
    }
```

Start delay measurement.

Step 5. Perform one of the following:

a. When you set the **oam performance-monitoring ip session stamp admin-state** command to **enable**, the proactive test starts immediately.

b. To start an on-demand session, use the **tools oam performance-monitoring ip session on-demand-action start** command.

Expected outcome

The on-demand session starts.

```
--{ + candidate shared default }--[  ]--
# tools oam performance-monitoring ip session session1 on-demand-action start
/oam/performance-monitoring/ip/session[session-name=session1]:
    OnDemand session started successfully
```

Stop delay measurement.

Step 6. Perform one of the following:

a. When you set the **oam performance-monitoring ip session stamp admin-state** command to **disable**, the proactive test stops immediately.

b. To stop an on-demand session, use the **tools oam performance-monitoring ip session on-demand-action stop** command.

Expected outcome

The on-demand session stops.

```
--{ + candidate shared default }--[  ]--
# tools oam performance-monitoring ip session session1 on-demand-action stop
/oam/performance-monitoring/ip/session[session-name=session1]:
    OnDemand session stopped successfully
```

c. You can use the **oam performance-monitoring ip session stamp test-duration** command to determine the duration of the test and the test stops after the completion of the configured time duration.

Example

Configure time duration for on-demand test

```
--{ +* candidate shared default }--[  ]--
A:srl1# info with-context oam performance-monitoring ip session session1 stamp test-
duration
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    stamp {
```

```
                        test-duration 60
                    }
                }
            }
        }
    }
```

5.2.2.3 Performing STAMP OAM-PM loss measurement

About this task

Perform the following steps to measure STAMP OAM-PM packet loss:

Procedure

Configure a loss events template.

Step 1. To configure a loss events template, use the **oam performance-monitoring ip loss loss-events-template** command and specify the loss event parameters as shown in the following example.

Example

Configuring a loss events template

```
--{ + candidate shared default }--[  ]--
A:srl1# info with-context oam performance-monitoring ip loss loss-events-template
 temp1
    oam {
        performance-monitoring {
            ip {
                loss {
                    loss-events-template temp1 {
        avg-flr-event forward {
                            raise-threshold 3.000
                            clear-threshold 0.000
                        }
                        hli-event aggregate {
                            raise-threshold 10
                            clear-threshold 0
                        }
                    }
                }
            }
        }
    }
```

Configure the session type.

Step 2. Perform one of the following:

a. To configure a proactive test session, use the **oam performance-monitoring ip session session-type proactive** command.

Example

Configuring a proactive test session

```
--{ +* candidate shared default }--[  ]--
A:srl1# info with-context oam performance-monitoring ip session session1 session-
type
```

```
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    session-type proactive
                }
            }
        }
    }
```

b. To configure an on-demand test session, use the **oam performance-monitoring ip session session-type on-demand** command.

Example

Configuring an on-demand test session

```
--{ +* candidate shared default }--[  ]--
# info with-context oam performance-monitoring ip session session1 session-type
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    session-type on-demand
                }
            }
        }
    }
```

Enable STAMP and configure loss measurement parameters.

Step 3. To enable STAMP and configure the delay measurement parameters, use the **info oam performance-monitoring ip session stamp** command and specify the parameters as shown in the following example.

Example

Configuring STAMP parameters for loss measurement

```
--{ + candidate shared default }--[  ]--
A:srl1# info with-context oam performance-monitoring ip session session1 stamp
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    stamp {
                        admin-state enable
                        test-id auto
                        interval 1s
                        delay {
                            bin-group gp1
                        }
                        loss {
                            flr-threshold 10
                            hli-force-count true
                            loss-event temp1
                            timing {
                                frames-per-delta-t 10
                                consecutive-delta-t 5
                                chli-threshold 2
                            }
                        }
                    }
                }
```

```
            }
        }
    }
```

Start loss measurement.

Step 4. Perform one of the following:

a. When you set the **oam performance-monitoring ip session stamp admin-state** command to **enable**, the proactive test starts immediately.

b. To start an on-demand session, use the **tools oam performance-monitoring ip session on-demand-action start** command.

Expected outcome

The on-demand session starts.

```
--{ + candidate shared default }--[  ]--
# tools oam performance-monitoring ip session session1 on-demand-action start
/oam/performance-monitoring/ip/session[session-name=session1]:
    OnDemand session started successfully
```

Stop delay measurement.

Step 5. Perform one of the following:

a. When you set the **oam performance-monitoring ip session stamp admin-state** command to **disable**, the proactive test stops immediately.

b. To stop an on-demand session, use the **tools oam performance-monitoring ip session on-demand-action stop** command.

Expected outcome

The on-demand session stops.

```
--{ + candidate shared default }--[  ]--
# tools oam performance-monitoring ip session session1 on-demand-action stop
/oam/performance-monitoring/ip/session[session-name=session1]:
    OnDemand session stopped successfully
```

c. You can use the **oam performance-monitoring ip session stamp test-duration** command to determine the duration of the test and the test stops after the completion of the configured time duration.

Example

Configure time duration for on-demand test

```
--{ +* candidate shared default }--[  ]--
A:srl1# info with context oam performance-monitoring ip session session1 stamp test-
duration
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    stamp {
                        test-duration 60
                    }
                }
            }
        }
```

```
    }
```

5.2.2.4 Displaying STAMP OAM-PM delay and loss measurement results

Procedure

To display the results of STAMP OAM-PM delay and loss measurement, use the following commands:

- **info from state oam performance-monitoring ip session** - See Displaying the results of delay and loss measurement.
- **info from state oam performance-monitoring ip session * | as table** - See Displaying OAM-PM IP session information with session-type
- **info from state oam performance-monitoring ip session * stamp delay measurement-result * index * | as table** - See Displaying OAM-PM IP session state detailed information with transmit and receive
- **info from state oam performance-monitoring ip session * stamp delay measurement-result * index * statistics bin-type * | as table**- See Displaying OAM-PM IP session delay statistics
- **info from state oam performance-monitoring ip session * stamp loss measurement-result * index * statistics | as table** - See Displaying OAM-PM IP session loss statistics

Example: Displaying the results of delay and loss measurement

The following example shows the results of delay and loss measurement.

```
--{ + candidate shared default }--[  ]--
# info from state with-context oam performance-monitoring ip session session1
    oam {
        performance-monitoring {
            ip {
                session session1 {
                    session-type proactive
                    destination-ip 192.0.2.1
                    destination-udp-port 862
                    source-ip 192.0.2.2

                    network-instance default
                    dscp CS6
                    profile in
                    ttl 255

                    measurement-interval 1-minute {
                        boundary-type clock-aligned
                        clock-offset 0
                        intervals-stored 32
                        threshold-alerts {
                            loss-event disable
                            delay-event disable
                        }
                    }
                    stamp {
                        admin-state enable
                        oper-state up
                        detected-tx-error none
                        test-id auto
                        test-id-in-use 2147483649
```

```
                        pad-tlv-size 0
                        interval 1s
                        statistics {
                            stamp-unrecognized-flag-received 0
                            stamp-malformed-flag-received 0
                        }
                        delay {
                            bin-group gp1
                            bin-group-binning active
                            measurement-result 1-minute {
                                newest-index 2
                                index 1 {
                                    oper-state in-progress
                                    suspect-status true
                                    start-time 2024-06-21T19:02:22.000Z
                                    elapsed-time 37
                                    statistics {
                                        frames-transmitted 37
                                        frames-received 37
                                        bin-type fd {
                                            forward {
                                                minimum 8
                                                maximum 10
                                                average 9
                                            }
                                            backward {
                                                minimum 8
                                                maximum 10
                                                average 9
                                            }
                                            round-trip {
                                                minimum 17
                                                maximum 20
                                                average 19
                                            }
                                            bin 0 {
                                                forward-measurements 37
                                                backward-measurements 37
                                                round-trip-measurements 37
                                            }
                                            bin 1 {
                                                forward-measurements 0
                                                backward-measurements 0
                                                round-trip-measurements 0
                                            }
                                        }

                                        bin-type ifdv {
                                            forward {
                                                minimum 0
                                                maximum 2
                                                average 1
                                            }
                                            backward {
                                                minimum 0
                                                maximum 3
                                                average 1
                                            }
                                            round-trip {
                                                minimum 0
                                                maximum 4
                                                average 1
                                            }
                                            bin 0 {
```

```
                                        forward-measurements 37
                                        backward-measurements 37
                                        round-trip-measurements 37
                                    }
                                    bin 1 {
                                        forward-measurements 0
                                        backward-measurements 0
                                        round-trip-measurements 0
                                    }
                                }
                            }
                        }

                    loss {
                        flr-threshold 10
                        hli-force-count true
                        loss-event temp1
                        timing {
                            frames-per-delta-t 10
                            consecutive-delta-t 5
                            chli-threshold 2
                        }
                        measurement-result 1-minute {
                            newest-index 2
                            index 1 {
                                oper-state in-progress
                                suspect-status true
                                start-time 2024-06-21T19:02:22.000Z
                                elapsed-time 37
                                statistics {
                                    frames-transmitted 37
                                    frames-received 37
                                    forward {
                                        out-loss 0
                                        available 3
                                        unavailable 0
                                        undetermined-available 0
                                        undetermined-unavailable 0
                                        high-loss-intervals 0
                                        consecutive-high-loss-intervals 0
                                        minimum-frame-loss-ratio 0
                                        maximum-frame-loss-ratio 0
                                        average-frame-loss-ratio 0
                                    }
                                    backward {
                                        in-loss 0
                                        available 3
                                        unavailable 0
                                        undetermined-available 0
                                        undetermined-unavailable 0
                                        high-loss-intervals 0
                                        consecutive-high-loss-intervals 0
                                        minimum-frame-loss-ratio 0
                                        maximum-frame-loss-ratio 0
                                        average-frame-loss-ratio 0
                                    }
                                }
                            }
                        }

}
```

Example: Displaying OAM-PM IP session information with session-type

The following example shows the OAM-PM IP session information.

```
--{ + candidate shared default }--[  ]--
# info from state oam performance-monitoring ip session * | as table
+------------+------------+------------+----------+------------+---------------+------------+
|  Session-  |  Session-  |   Stamp    |  Stamp   |   Stamp    |   Stamp       |   Stamp    |
|    name    |    type    |admin state |oper-state| detected-  |stamp session- | delay bin- |
|            |            |            |          |  tx-error  | identifier    |   group    |
+============+============+============+==========+============+===============+============+
| session1   | proactive  | enable     |    up    | none       |    1005       | gp1        |
|            |            |            |          |            |               |            |
+------------+------------+------------+----------+------------+---------------+------------+
```

Example: Displaying OAM-PM IP session state detailed information with transmit and receive

The following example shows the OAM-PM IP session state detailed information with transmit and receive.

```
--{ + candidate shared default }--[  ]--
# info from state oam performance-monitoring ip session * stamp delay measurement-result
* index * | as table
+---------+---------+-----+---------+--------+---------------+-------+-----------+------------+
| Session | Measure |Index|  Oper-  |Suspect | Start time    |Elapsed|Statistics | Statistics |
|         |  ment-  |     |  state  |-status |               | -time |frames     | frames-    |
|         | result  |     |         |        |               |       |transmitted| received-  |
+=========+=========+=====+=========+========+===============+=======+===========+============+
|session 1|1- minute|  21 |completed| false  |2025-01-29T20:4|  60   |    0      |    0       |
|         |         |     |         |        |7:00.00 OZ     |       |           |            |
|session 1|1- minute|  22 |completed| false  |2025-01-29T20:4|  60   |    0      |    0       |
|         |         |     |         |        |8:00.00 OZ     |       |           |            |
|session 1|1- minute| 23  |completed| false  |2025-01-29T20:4|  60   |    0      |    0       |
|         |         |     |         |        |9:00.00 0Z     |       |           |            |
+---------+---------+-----+---------+--------+---------------+-------+-----------+------------+
```

Example: Displaying OAM-PM IP session delay statistics

The following example shows the OAM-PM IP session delay statistics.

```
--{ + candidate shared default }--[  ]--
# info from state oam performance-monitoring ip session * stamp delay measurement-result
* index * statistics bin-type * | as table
+---------+------+------+------+------+------+------+------+------+------+------+------+------+
| Session | Meas | Index| Bin- | Forw | Forw | Forw | Back | Back | Back | Roun | Roun | Roun |
|         | urem |      | metr | ard  | ard  | ard  | ward | ward | ward |  d-  |  d-  |  d-  |
|         | ent- |      |  ic  | mini | maxi | aver | mini | maxi | aver | trip | trip | trip |
|         | resu |      |      | mum  | mum  | age  | mum  | mum  | age  | mini | maxi | aver |
|         |  lt  |      |      |      |      |      |      |      |      | mum  | mum  | age  |
+=========+======+======+======+======+======+======+======+======+======+======+======+======+
|session 1| 1-   |    1 | fdr  |    0 |    0 |    0 |    0 |    0 |    0 |    0 |    0 |    0 |
|         | minu |      |      |      |      |      |      |      |      |      |      |      |
|         | te   |      |      |      |      |      |      |      |      |      |      |      |
|session 1| 1-   |    1 | ifdv |    0 |    0 |    0 |    0 |    0 |    0 |    0 |    0 |    0 |
|         | minu |      |      |      |      |      |      |      |      |      |      |      |
|         | te   |      |      |      |      |      |      |      |      |      |      |      |
|session 1| 1-   |    2 | fdr  |    0 |    0 |    0 |    0 |    0 |    0 |    0 |    0 |    0 |
|         | minu |      |      |      |      |      |      |      |      |      |      |      |
|         | te   |      |      |      |      |      |      |      |      |      |      |      |
+---------+------+------+------+------+------+------+------+------+------+------+------+------+
```

Example: Displaying OAM-PM IP session loss statistics

The following example shows the OAM-PM IP session loss statistics.

```
--{ + candidate shared default }--[  ]--
# info from state oam performance-monitoring ip session * stamp loss measurement-result * index *
 statistics | as table
+-------+--------+-----+-------+-------+----+-------+-------+-------+-----+-----+-----+-------+
|Session|Measu-  |Index|Frames-|Frames-|For-|Forward|Forward|Forward|Back-|Back-|Back-|Back-  |
|       |rement- |     |transm-|recei- |ward|minimum|maximum|average|ward-|ward-|ward |ward   |
|       |result  |     |itted  |ved    |out |frame  |frame  |frame  |in-  |min  |max  |average|
|       |        |     |       |       |loss|loss   |loss   |loss   |loss |frame|frame|frame  |
|       |        |     |       |       |    |ratio  |ratio  |ratio  |     |loss |loss |loss   |
|       |        |     |       |       |    |       |       |       |     |ratio|ratio|ratio  |
+=======+========+=====+=======+=======+====+=======+=======+=======+=====+=====+=====+=======+
| ip-1  |1-minute| 1   |    37 |    30 |  0 |     0 |     0 |     0 |   0 |   0 |   0 |     0 |
| ip-1  |1-minute| 2   |    60 |    60 |  0 |     0 |     0 |     0 |   0 |   0 |   0 |     0 |
| ip-1  |1-minute| 3   |    60 |    60 |  0 |     0 |     0 |     0 |   0 |   0 |   0 |     0 |
| ip-1  |1-minute| 4   |    60 |    60 |  0 |     0 |     0 |     0 |   0 |   0 |   0 |     0 |
| ip-1  |1-minute| 5   |    60 |    60 |  0 |     0 |     0 |     0 |   0 |   0 |   0 |     0 |
| ip-1  |1-minute| 6   |    60 |    60 |  0 |     0 |     0 |     0 |   0 |   0 |   0 |     0 |
+-------+--------+-----+-------+-------+----+-------+-------+-------+-----+-----+-----+-------+
```

6 Service Activation Testhead (SAT)

Note: This feature is supported on 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, 7250 IXR-X3b, and 7250 IXR Gen 3 platforms.

Introduction

SAT is a point-to-point out-of-service test methodology defined in the ITU-T Y.1564 standard. It is used to validate Ethernet-based service configuration and performance before the service is delivered to a customer. The test ensures that the network can support the service in accordance with the agreed-upon service level agreements (SLAs).

ITU-T Y.1564 defines two classes of tests:

- Service configuration tests: These are short-duration tests that verify the accuracy of service configuration. They include:
  - committed information rate (CIR)
  - CIR combined with peak information rate (CIR-PIR)
  - policing (traffic is tested at 125% of the configured PIR)
- Service performance test: This typically long-duration test evaluates key performance metrics of the service.

The SAT tool administers tests based on configured transmission rates and measures performance metrics, including:

- throughput
- frame loss ratio (FLR)
- frame delay (FD), minimum, maximum, and average
- frame delay variation (FDV), minimum, maximum, and average

Additional statistics include the number of transmitted and received packets, start and end times, and remaining test duration. Statistics are available during the test execution.

SAT architecture

SAT relies on ITU-T Y.1731 ETH-CFM and MEPs. The protocols required to execute the test are:

- Ethernet loopback messages (ETH-LBM) - Throughput streams are encapsulated in ETH-LBM.
- Ethernet delay measurement messages (ETH-DMM) - Delay measurements are gathered using ETH-DMM.

MEPs must be configured on the subinterface endpoints. The target MEP that reflects the test frames back to the source must process ETH-LBM without control plane assistance and respond appropriately with ETH-LBR frames. Because ETH-DMM frames arrive on the target every 500 ms, control plane assistance may be used.

Note: Platform-specific configuration requirements of the reflecting MEP must be reviewed before enabling SAT. Different platforms may require different reflector configurations.

The total bandwidth available on the platform is shared with other protocols, such as BFD and ETH-CCM. Therefore, avoid running SAT traffic at the platform's maximum rate when BFD or ETH-CCMs is also active.

SAT considerations

When testing between the Layer 2 subinterfaces within a MAC-VRF, traffic other than SAT traffic cannot be sent to or from the subinterfaces under test. Furthermore, the combined traffic exchanged between the subinterfaces in the MAC-VRF and the test streams must not exceed the overall capability of the network instance, or the network bandwidth. When traffic is generated from an up MEP in a MAC-VRF, the MAC address of the peer must be learned on the correct egress subinterface before starting the test. Broadcast, unknown, and multicast (BUM) traffic is likely to invalidate test results because of its distribution to all points in network instance.

The SAT tool is not intended to be used during active network events. Running a service activation test during such events can worsen ongoing events.

These are Layer 2 test functions. If the parent interface is a Layer 3 subinterface, as is the case with network instance default and IP-VRF, the state of the IP interface does not impact the test.

The SAT OAM tool does not verify the operational state of the service or subinterface before starting a test. You need to ensure that both the service and the subinterface are up before initiating the test. If either is down, the SAT tool reports a failure.

Changes to the service test configuration can only be made when the testhead tool is administratively disabled. You cannot change the administrative state if an associated service test is currently running. The service test must have completed or need to be stopped before any configuration changes are allowed.

These tests are point-to-point between a pair of MEPs. Only unicast traffic flows should be tested with the testhead OAM tool.

All test statistics are stored in volatile YANG state memory and the data must be collected immediately after the test completes.

The testhead tool operates using the Layer 2 rate, which excludes Layer 1 overhead. This overhead includes inter-frame gap (IFG) and preamble. Together, they add approximately 20 bytes (IFG is 12 bytes and preamble is 8 bytes) to each Ethernet frame. It is important to consider the percentage overhead based on the selected frame size used by the stream.

Disruptive fault conditions such as a MEP going down, a subinterface failure, or a hardware or card reboot, may stop a running test.

Deleting the **service-test** configuration, the **oam service-activation-testhead** container, or the **oam** container will delete all associated **service-test** statistics. A rollback has the same effect and will delete all associated statistics. Rollback will restore only the configuration and not the service-tests or their statistics. Test run numbers will be reinitialized to one.

6.1 SAT launch point

The following are the MEP, subinterface, and network instance guidelines and limitations for performing the service test:

- Each subinterface reference must be unique across all test streams within a service test. The MEP references must also be unique per stream.

- Within a single service test, all streams must use the same subinterface type; either Layer 2 or Layer 3 subinterfaces. Mixing Layer 2 and Layer 3 subinterfaces within the same test is not allowed.
- Subinterfaces configured over Link Aggregation Groups (LAGs) are not supported.
- The subinterfaces can be:
  - untagged
  - single-tagged
  - double-tagged
- When a test is launched from an Up MEP on a multi-complex system such as, the 7250 IXR-6e,7250 IXR-10e, or 7250 IXR-X3b, the same complex must be used for both ingress and egress. If ingress and egress are on different complexes, the delay measurements are not accurate.

The following table describes the supported combinations of network instance types, MEPs, and subinterfaces required to perform the test.

Table 13: Platform, network instance, and subinterface support

Platforms	Network instance and subinterface type				Ethernet interfaces
	MAC VRF L2 Subinterface	VPWS L2 Subinterface	IP-VRF L3 subinterface Down MEP only	default L3 subinterface Down MEP only	
7730 SXR	No	No	No	No	No
7250 IXR Gen 2c+	Yes	No	Yes	Yes	No
7250 IXR-X1b	Yes	No	Yes	Yes	No
7250 IXR-X3b	Yes	No	Yes	Yes	No
7250 IXR Gen 3	No	No	Yes	Yes	No

6.2 SAT configuration

Acceptance criteria template

The acceptance criteria template is used to configure the thresholds that indicate the acceptable range for the service performance metrics. At the end of the test, the measured values for FD, FDV, and FLR are compared against the configured thresholds to determine the pass or fail criteria and to generate a trap to the management station.

You can configure the following parameters using the acceptance criteria template:

- CIR threshold
- PIR threshold
- Delay threshold
- Delay variation threshold
- Loss threshold

- Loss threshold policing
- m-factor

You can configure the m-factor (margin factor) to take into account the variances in the measured throughput and the configured throughput. The **default** acceptance criteria applies no thresholds—all tests automatically pass when run to completion. This **default** acceptance criteria cannot be modified and is viewable only using the **info from state oam service-activation-testhead acceptance-criteria-template default** command.

Frame size template

The service testhead tool can be configured to generate a flow containing a frame sequence of different sizes. The testhead tool generates packet sizes as specified in the frame size template. The template is used to specify a mix of frames with different sizes that the tool can generate and choose from. The following table lists the frame size defaults for the default template. The **default** frame size template cannot be modified, and is viewable only using the **info from state oam service-activation-testhead frame-size-template default** command.

Table 14: Ethernet frame sizes and size designations

a	b	c	d	e	f	g	h	u
64	128	256	512	1024	1280	1518	9212	2000

The Ethernet frame size ranges from 64 to 9212 bytes, and the designated letters are used to specify the frame size. Frame size is configured using the **sequence** command in the **frame-mix** context in a service stream. The frame mix allows the configuration of only a single, fixed frame size sequence. The configured size applies to all frames in the test flow. If the frame size exceeds the interface MTU, packets are silently dropped.

Note:
As part of the test, frame size configuration does not include C-tags or S-tags, which, if present, augment the frame size of the Ethernet frame. For example:

- for single tagged subinterface, the throughput calculation must include an additional 4 bytes
- for double tagged subinterface, the throughput calculation must include an additional 8 bytes

Service test

The **service-test** is the main test container that includes all the test elements. Every element must pass for the overall service test to complete with an operational state of `pass`.

Up to eight service tests can be active at any time on each line card complex.

For Layer 2 network instances, multiple streams are allowed. Each test stream must have a unique source MEP over different Layer 2 subinterfaces. For Layer 3 subinterface, only one test stream is supported .

The stream run types are sequential and parallel. Sequential, processes all streams of the same type one after another before moving on to the next type. Parallel, runs all streams of the same type simultaneously. Performance tests always run concurrently regardless of the stream run type.

You can configure the test durations for CIR, CIR-PIR, policing, and performance types for up to 24 hours, 59 minutes, and 59 seconds. Performance metrics are measured after the test reaches the target rate. You can check the status and progress of the test at any time. Minor discrepancies in start and stop timing may occur, affecting throughput calculations. For long-duration tests, this impact is typically negligible.

Service stream

Up to eight service streams can be active at any time on each line card complex. Each stream is associated with an acceptance criteria template that contains thresholds to indicate the acceptable range for the service performance metrics. Each stream is associated with a frame size template that contains the Ethernet frame size ranges from 64 bytes to 9212 bytes, and the designated letters that are used to specify the frame size. The frame size for the specific service stream is configured using the **sequence** command in the **frame-mix** context in a service stream. The frame mix only allows the configuration of a single fixed frame size sequence. The configured size applies to all frames in the test flow.

You must configure a source MEP, which is associated with a subinterface and network instance under the test, and a destination MEP MAC address for the ETH-CFM LBM frame payload configuration. The data pattern for the payload, dot1p, and discard eligibility indicator (DEI) for the VLAN tags can also be specified. You cannot configure the VLAN tag; it is derived from the subinterface hosting the MEP. The MEP reference must be unique across all streams. The test fails to start if the MEP is not administratively enabled.

The testhead OAM tool generates traffic up to the specified rate and measures service performance metrics such as delay, delay variation, and loss.

Test start

Use the **tools oam service-activation-testhead start** command to start the test.

The following are the guidelines to start a test:

- At least one service stream must be enabled to start a service test.
- At least one **test-type** must be configured to administratively enable the service stream.
- At least one **test-duration** must meet the minimum test duration to administratively enable the service test and service stream. Every test has a minimum test duration of 30 seconds.
- The service test must be admin enabled before issuing the start command.
- The associated interface must be operationally up.
- The MEP must be admin enabled.
- A unicast destination MAC address must be configured.
- The total number of stored service test records must be less than 500.
- The bandwidth required for the test is not available on the line card.

6.3 Configure acceptance criteria template

Procedure

To configure acceptance criteria template, use the **oam service-activation-testhead acceptance-criteria-template** command. Provide a name for the template and configure the parameters shown in the example.

Example: Acceptance criteria template configuration

```
--{ + candidate shared default }--[  ]--
# info with-context oam service-activation-testhead acceptance-criteria-template test-1
    oam {
        service-activation-testhead {
            acceptance-criteria-template test-1 {
```

```
                description None
                cir-threshold 10000000
                pir-threshold 1000000
                loss-threshold 0.0500
                loss-threshold-policing 25.0000
                delay-threshold 100
                delay-var-threshold 20
                m-factor 1000
            }
        }
    }
```

6.4 Configure frame size template

Procedure

Use the **oam service-activation-testhead frame-size-template** command and provide a name for the template. Configure the parameters shown in the example.

Example: Frame size template configuration

```
--{ + candidate shared default }--[  ]--
# info with-context oam service-activation-testhead frame-size-template test-1
    oam {
        service-activation-testhead {
            frame-size-template test-1 {
                size-a 64
                size-b 128
                size-c 256
                size-d 512
                size-e 1024
                size-f 1280
                size-g 1518
                size-h 9212
                size-u 2000
            }
        }
    }
```

6.5 Configure SAT and display results

Prerequisites

Ensure that:

- the MEP is admin enabled.
- the interface is operationally up.

About this task

Perform the following steps to configure the SAT service tests and service streams:

Procedure

Step 1. Execute the **oam service-activation-testhead service-test** command to configure the service test. Configure the parameters as shown in the example.

Example

Configuring the service test parameters

```
--{ + candidate shared default }--[  ]--
# info with-context oam service-activation-testhead service-test SAT-1
    oam {
        service-activation-testhead {
            service-test sampletest-1 {
                description None
                stream-run-type parallel
                test-duration {
                    cir {
                        minutes-seconds 05:00
                    }
                    cir-pir {
                        minutes-seconds 10:00
                    }
                    policing {
                        minutes-seconds 10:00
                    }
                    performance {
                        hours-minutes-seconds 00:15:00
                    }
                }
            }
        }
    }
```

Step 2. Execute the **oam service-activation-testhead service-test service-stream** command to configure the service stream. Configure the parameters as shown in the example.

Example

Configuring the service stream parameters

```
--{ + candidate shared default }--[  ]--
# info with-context oam service-activation-testhead service-test sampletest-1 service-
stream 1
    oam {
        service-activation-testhead {
            service-test sampletest-1 {
                service-stream 1 {
                    description None
                    acceptance-criteria-template test-1
                    frame-mix {
                        frame-size-template test-1
                        sequence e
                    }
                    test-types {
                        cir true
                        cir-pir true
                        policing true
                        performance true
                    }
                    frame-payload {
                        ethernet {
                            dst-mac 02:02:03:01:01:01
                            eth-cfm {
```

```
                                source {
                                    domain-id 10
                                    association-id 10
                                    mep-id 1
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

Step 3. Configure the administrative state of the service stream as enable and then configure the administrative state of the service test as enable.

Example

Administratively enabling the service stream and service test

```
--{ + candidate shared default }--[  ]--
# info with-context oam service-activation-testhead
    oam {
        service-activation-testhead {
            acceptance-criteria-template test-1 {
                description None
                cir-threshold 10000000
                pir-threshold 1000000
                loss-threshold 0.0500
                loss-threshold-policing 25.0000
                delay-threshold 100
                delay-var-threshold 20
                m-factor 1000
            }
            frame-size-template test-1 {
                size-a 64
                size-b 128
                size-c 256
                size-d 512
                size-e 1024
                size-f 1280
                size-g 1518
                size-h 9212
                size-u 2000
            }
            service-test sampletest-1 {
                admin-state enable
                description None
                stream-run-type parallel
                test-duration {
                    cir {
                        minutes-seconds 05:00
                    }
                    cir-pir {
                        minutes-seconds 10:00
                    }
                    policing {
                        minutes-seconds 10:00
                    }
                    performance {
                        hours-minutes-seconds 00:15:00
                    }
                }
                service-stream 1 {
```

```
                    admin-state enable
                    description None
                    acceptance-criteria-template test-1
                    frame-mix {
                        frame-size-template test-1
                        sequence e
                    }
                    test-types {
                        cir true
                        cir-pir true
                        policing true
                        performance true
                    }
                    frame-payload {
                        ethernet {
                            dst-mac 02:02:03:01:01:01
                            eth-cfm {
                                source {
                                    domain-id 10
                                    association-id 10
                                    mep-id 1
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

Step 4. Use the **tools oam service-activation-testhead service-test start** command to start a service test.

Example

Starting a service test

```
--{ + candidate shared default }--[  ]--
# tools oam service-activation-testhead service-test sampletest-1 start
/oam/service-activation-testhead/service-test[test-name=sampletest-1]:
    Service test sampletest-1 is started with run index 3
```

Note: The following guidelines for starting a service test apply:

- You cannot start two service tests with the same name. When the **start** command is issued, attempting to start a second service test with the same name fails the **start** command with an appropriate error.
- When a **stop** command is issued for a service test that is not running, an error message indicating the test is not running appears.

Step 5. Use the **show oam service-activation-testhead service-activation-tests** command to display the run instance summary of a service test.

Example

Displaying the run instance summary

```
--{ + candidate shared default }--[  ]--
# show oam service-activation-testhead service-activation-tests
+--------------+----------+------------------------------+
|  test-name   | test-run |          completed           |
```

```
+==============+==========+============================+
| sampletest-1 | 1        | 2025-07-04T21:44:00.0Z+05:30 |
| sampletest-1 | 2        | 2025-07-04T21:44:00.0Z+05:30 |
| sampletest-1 | 3        | 2025-07-04T21:44:00.0Z+05:30 |
| sampletest-2 | 1        | 2025-07-04T21:44:00.0Z+05:30 |
| sampletest-2 | 2        | 2025-07-04T21:44:00.0Z+05:30 |
| sampletest-2 | 3        | 2025-07-04T21:44:00.0Z+05:30 |
| sampletest-3 | 1        | 2025-07-04T21:44:00.0Z+05:30 |
| sampletest-3 | 2        | 2025-07-04T21:44:00.0Z+05:30 |
| sampletest-3 | 3        | 2025-07-04T21:44:00.0Z+05:30 |
| test-1       | 1        |                            |
+--------------+----------+----------------------------+
```

Step 6. Use the **show oam service-activation-testhead service-test run** command to display the service test results.

Example

Displaying the service test results

```
--{ + candidate shared default }--[  ]--
# show oam service-activation-testhead service-test sampletest-1 run 1
-------------------------------------------------------------------------------
Y.1564 Results for run 1
Service Test sampletest-1
-------------------------------------------------------------------------------
Description       : None
Oper State        : passed
Start Time        : 2025-07-04T16:47:07.123Z
End Time          : 2025-07-04T16:47:44.610Z
Stream Run Type   : parallel
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
Y.1564 Results for Stream 1 Run 1
Service Test sampletest-1
-------------------------------------------------------------------------------
Description  : None
Oper State   : passed
Network Inst : default
Source MEP   : 52                   MEP Domain   : 10
MEP Assoc.   : 10                  Subinterface : ethernet-1/1.1
Source MAC   : C0:14:B8:0D:8B:BA     Dest MAC     : 02:02:03:01:01:01
Pattern (Dec): 0                     Pattern (Hex) : 0x0
C-Tag Dot1p  : 7  C-Tag DEI    : false
S-Tag Dot1p  : 7  S-Tag DEI    : false
Frm Size Tmpl: fst
Frm Sequence : c                     Frame Size a : 64
Frame Size b : 150                   Frame Size c : 256
Frame Size d : 512                   Frame Size e : 1024
Frame Size f : 1600                  Frame Size g : 1518
Frame Size h : 9212                  Frame Size u : 2000
Acp Crit Tmpl: act
Test Types   : CIR-PIR
-------------------------------------------------------------------------------


-------------------------------------------------------------------------------
Y.1564 Results for Stream 1 Run 1 Test Type cir-pir
Service Test sampletest-1
-------------------------------------------------------------------------------
Oper State   : passed
Test Duration: 00:30 (mm:ss)            Time Left    : 0 s
Start Time   : 2025-07-04T16:47:09.489Z
```

```
End Time     : 2025-07-04T16:47:39.595Z
Frms Injected: 2939470                  Frms Received: 2939470
Min Delay    : 15 us                    Min Delay Var: 0 us
Max Delay    : 17 us                    Max Delay Var: 2 us
Avg Delay    : 15 us                    Avg Delay Var: 0 us
-------------------------------------------------------------------------------

===============================================================================
Test Compliance Report
===============================================================================
Criteria       Throughput(kbps)    FLR (%)        Delay (us)     Delay Var (us)
-------------------------------------------------------------------------------
Acceptable     200000              (Not Specified)(Not Specified)(Not Specified)
Configured     200000              (Not Specified)(Not Specified)(Not Specified)
M-Factor       2000                N/A            N/A            N/A
Measured       199958              0.0000         15             0
Result         pass                pass           pass           pass
===============================================================================
```

7 OAM protocol interactions

This section describes interactions between like OAM style protocols

7.1 Service Activation Testhead and Packet Link Qualification

The Service Activation Testhead (SAT) is an out-of-service test used to qualify service configuration and performance metrics, including throughput, delay, and packet loss, across various test types. The Packet Link Qualification (PLQ) is an out-of-service physical interface test that verifies the integrity of the physical link. SAT and PLQ cannot coexist on the same complex, as they are disruptive and block each other.

If SAT is actively running on a complex and PLQ is configured on the same complex (using either an ASCI loopback or packet generator), there are two possible outcomes:

- Shared interface resource: If SAT and PLQ require the same interface resource, SAT fails with a service test operational state of stopped-by-fault. This occurs because configuring PLQ for an interface sets the interface operational state to testing, a non-operational state that causes SAT to fail.
- Different interface resources: If SAT and PLQ use different interface resources, SAT continues to run, but attempting to start the PLQ test fails the precondition with the status message: "Test resource not available". This happens because SAT is already consuming the complex resources.

Similarly, if PLQ is configured on an interface (via ASCI loopback or packet generator), it reserves the complex resources, preventing SAT from running successfully:

- Same interface resource: If SAT requires the same interface as the PLQ configuration, it fails to start and produce the error: "Error in /oam/service-activation-testhead/service-test(test-name=test): subinterface is operationally down for stream"
- Different interface resources: If the interfaces differ, SAT attempts to start but fails with a service test operational state of stopped-by-fault, because PLQ is still consuming the complex resources.

8 sFlow

sFlow is used to monitor data traffic flows traversing different points in a network. The sFlow functionality consists of an sFlow agent and one or more sFlow collectors. The agent is software that runs on a network element. It samples and reports flow headers and statistics. The collector is software that typically runs on a remote server. It receives flow headers and statistics from one or more sFlow agents.

Sampling and reporting are accomplished as the sFlow agent running on a network element takes periodic samples of ingress traffic and reports the data to the configured collectors. The network element does not maintain a local flow cache; instead, the sampled header information is immediately sent to the collector without additional processing. When samples are sent, the sFlow sample sequence number is generated for each source interface, ensuring correct ordering and correlation of samples at the collector. sFlow sample sequence number per source interface is supported in IPv6 collectors on 7250 IXR Gen 2c+, 7250 IXR Gen 3, and 7250 IXR-X4 platforms.

SR Linux supports sFlow version 5 behavior and formats. On 7250 IXR chassis-based systems, sFlow is implemented in hardware. On 7220 IXR systems, sFlow functionality is implemented in software. sFlow behavior is identical on both platforms, with the following exceptions:

- Frame sample sizes of 256 or 512 bytes are supported only on 7250 IXR systems.
- IPv6 sFlow collector configuration is supported only on 7250 IXR systems.
- IPv6 UDP checksum configuration is supported only on 7250 IXR Gen 2c+ systems.

sFlow uses a single source ID per interface to identify the source of sampled traffic. The source ID is derived from the ingress or egress interface, based on the configured sampling direction. The interface index (ifindex) is structured such that the physical Ethernet interface identifier occupies the lower 16 bits, while the subinterface index is placed in the upper bit positions. This structure ensures that physical interface identifiers fit within the sFlow source ID range and aligns with sFlow and SNMP interface identification requirements.

Note:
On 7250 IXR Gen 3 platforms, an interface cannot simultaneously support sFlow egress sampling and act as a source for egress mirroring. If both are configured, the system rejects the configuration.

To enable sFlow egress sampling on an interface, you must first remove the interface as an egress mirror source. Conversely, to configure an interface for egress mirroring, you must first remove the sFlow configuration from that interface.

8.1 sFlow sampling

sFlow works by sampling flow data and reporting the samples to the configured Flow collectors. Based on the configured system sampling rate, the forwarding plane samples ingress packet flows and sends the sampled headers to the Flow agent in the control plane.

All ingress packets are subject to sampling. By default, 256 bytes are sampled from each packet. Each sample includes the following:

- 7220 IXR systems – samples include the top 256 bytes of the sampled packet, starting at the outer Ethernet header
- 7250 IXR systems – samples include the top 256 or 512 bytes of the sampled packet, starting at the outer Ethernet header

The sampled packets are sent to the configured sFlow collectors using the sFlow raw packet data format. For sampled IPv4 packets, IPv4 header data fields are included with the raw data. For sampled IPv6 packets, IPv6 header data fields are included with the raw data.

sFlow DSCP settings are as follows:

- A default DSCP value of 0 is assigned to flow and counter samples.
- You can change the DSCP value in the sFlow configuration
- The DSCP value applies to all collectors.

8.2 IPv6 UDP checksum

Note: This feature is supported on 7250 IXR Gen 2c+ platforms.

You can configure the checksum value used within sFlow sample messages sent to an IPv6 collector on 7250 IXR Gen 2c+ systems using the **system sflow ipv6-udp-checksum** command. The default IPv6 checksum value is set to **all-ones**. The configurable values are **zero** and **all-ones**. When you set the value to **zero**, the UDP checksum is set to 0x0 and when you set the values to **all-ones**, the UDP checksum is set to 0xFFFF.

8.3 Egress sFlow sampling on 7220 IXR-H4 platforms

Egress sampling is done at the egress VoQ (Virtual Output Queue) of the ingress forwarding chip on 7220 IXR-H4 platforms. The sample rates are separately configured for ingress and egress traffic for each sFlow enabled port.

The following considerations apply for egress sFlow sampling:

- Egress sFlow sampling is performed on each port, and not on a LAG or a subinterface.
- Because egress sampling is done at the egress VoQ of the ingress forwarding chip, true egress samples are not obtained. For example, tunneling information is not captured accurately.
- The rate at which sFlow samples are collected and exported is restricted by CPU capacity.
- Packets injected into the network by the CPU are not included in the sFlow sampling.
- Samples include the first 256 bytes of each sampled packet which are received from XDP (eXtensible Data Path) as raw header type.
- No extended data formats are used, and detailed metadata about the packets beyond basic header information is not included in the samples.
- When a packet is sampled on ingress, the same packet cannot be sampled for egress because of a BCM limitation.

8.4 sFlow collector reporting

sFlow reports sampled headers and statistics to the configured collectors using IP/UDP datagrams. UDP port 6343 is the default destination port, but you can optionally configure a different port. Sampled packets are sent as soon as the samples are taken, and interface statistics are sent at 10 second intervals. SR Linux supports up to eight remote IPv4 sFlow collectors or one remote IPv6 sFlow collector. IPv6 sFlow collector configuration is supported only on 7250 IXR systems. IPv4 and IPv6 sFlow collectors are mutually exclusive and cannot be configured simultaneously. Each collector can only have one IPv4 or IPv6 address. The flow and counter samples are aggregated in an sFlow datagram packet in software implementation.

8.5 sFlow counter samples

Another aspect of the sFlow agent is streaming of interface statistics to configured sFlow collectors. Statistics are only sent to a collector if sFlow has been enabled on an interface. Interface statistics are sent based on a default poll-interval of 10 seconds with a separate timer for each interface. When the interval expires. the current value of each associated statistics are sent to the configured collectors.

The interface counter sample contains:

- Interface index
- Interface type
- Interface speed
- Oper and admin status
- Input octets
- Input packets
- Input broadcast packets
- Input discards packets
- Output errors
- Output octets
- Output packets
- Output broadcast packets
- Output discards packets

8.6 Configuring the sFlow agent

Procedure

To configure the sFlow agent on the system, you enable sFlow, and optionally configure the sampling rate (by default, 1 out of every 10 000 packets) and sample size (by default, 256 bytes are sampled from each packet).

Example: Configuring the sFlow agent

The following example enables sFlow on the system and configures the system sampling rate and sample size. The polling interval is not configurable. The following sample size options apply:

- 7220 IXR-D2, 7220 IXR-D3, 7220 IXR-D4, 7220 IXR-D5, and 7220 IXR-H systems: 256 bytes
- 7250 IXR-6, 7250 IXR-10, 7250 IXR-6e, 7250 IXR-10e, and 7250 IXR-X3b systems: 256 or 512 bytes

```
--{ + candidate shared default }--[  ]--
# info with-context system sflow
    system {
        sflow {
            admin-state enable
            source-address 10.0.0.1
            sample-rate 50
            dscp CS6
        }
    }
```

8.7 Configuring IPv6 UDP checksum

Procedure

Note: This feature is supported on 7250 IXR Gen 2c+ platforms.

To configure the IPv6 checksum value on 7250 IXR Gen 2c+ systems, use the **system sflow ipv6-udp-checksum** command.

Example: Configure IPv6 UDP checksum

This example configures the checksum value used within the sFlow sample messages on 7250 IXR Gen 2c+ systems.

```
--{ + candidate shared default }--[  ]--
# info with-context system sflow
    system {
        sflow {
            ipv6-udp-checksum zero
        }
    }
```

8.8 Configuring sFlow collectors

Procedure

The sFlow agent sends sampled packets to sFlow collectors. You can configure up to eight IPv4 sFlow collectors or one IPv6 sFlow collector to receive the data. IPv6 sFlow collector configuration is supported only on 7250 IXR systems. IPv4 and IPv6 sFlow collectors are mutually exclusive and cannot be configured simultaneously.

To configure an sFlow collector, you specify its IP address, associated network instance, and IP address to be used as the source IP address in sFlow packets sent from SR Linux to the collector. You can optionally specify a destination port (by default, this is UDP port 6343).

Note:

- Configuring a network instance is mandatory.
- A collector cannot be reached using the **mgmt** network-instance.
- A collector IP address cannot reside in a local subnet. Although the configuration succeeds, the system ignores it. Create a static route to the collector subnet and configure ARP (static or dynamic) for the next hop of the static route.

Example: Configuring IPv4 sFlow collectors

The following example configures two IPv4 sFlow collectors. The IP address for each collector is configured, as well as its network instance and source IP address. Each collector receives all samples. The collector DSCP value for flow samples is also configured. If no value is specified, the default DSCP value of 0 applies.

```
--{ * candidate shared }--[  ]--
#info with-context system sflow
    system {
        sflow {
            dscp 14
            collector 1 {
                collector-address 10.50.4.1
                source-address 192.0.2.1
                network-instance default
            }
            collector 2 {
                collector-address 10.50.4.2
                source-address 10.1.5.2
                network-instance default
                port 4310
            }
        }
    }
```

Example: Configuring an IPv6 sFlow collector

The following example configures one IPv6 sFlow collector. The IP address for the collector is configured, as well as its network instance and source IP address. The collector receives all samples.

Note: Only one IPv6 collector with a **collector** value of 1 can be configured.

```
--{ * candidate shared default }--[  ]--
# info with-context system sflow
    system {
        sflow {
            collector 1 {
                collector-address 2001:db8::1
                network-instance default
                source-address 2001:db8::2
            }
        }
    }
```

8.9 Configuring sFlow for an interface

Procedure

When sFlow is configured for an Ethernet or a LAG interface, the ingress packets are taken for sampling according to the **sample-rate**.

The following considerations apply for sFlow on a LAG interface:

- sFlow on LAG feature is available on 7250 IXR-6, 7250 IXR-10, 7250 IXR-6e, 7250 IXR-10e, 7250 IXR-X1b, and 7250 IXR-X3b platforms.
- When sFlow on LAG interface is disabled, the sFlow state of the member ports are also disabled.
- When sFlow on LAG interface is enabled, the sFlow state of the member ports follow the individual sFlow admin state that is configured. The default value is **enable**.
- The **Input interface** field in the flow samples of the ingress traffic collected on the LAG port displays the **ifIndex** of the LAG member port.
- The **Output interface** field in the flow samples of the ingress unicast traffic that is marked as egress via a LAG port displays the **ifIndex** of the LAG port.

Example: Configuring sFlow for an Ethernet interface

The following example enables sFlow on an Ethernet interface.

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

Example: Configuring sFlow for a LAG interface

The following example enables sFlow on a LAG interface.

```
--{ + candidate shared default }--[  ]--
# info with-context interface lag1
    interface lag1 {
        sflow {
            admin-state enable
        }
        lag {
            lag-type static
            min-links 2
        }
    }
```

8.10 Configuring sFlow on 7220 IXR-H4 platforms

Procedure

To configure sFlow on 7220 IXR-H4 platforms, you enable sFlow on an interface and specify the sample rates for ingress and egress traffic.

Example: Configuring sFlow on a 7220 IXR-H4 interface

The following example enables sFlow on an interface and configures ingress and egress sampling rates.

```
--{ + candidate shared default }--[  ]--
# info with-context interface ethernet-1/1
    interface ethernet-1/1 {
        admin-state enable
        sflow {
            admin-state enable
            ingress-sampling-rate 562
            egress-sampling-rate 256
        }
    }
```

8.11 Displaying the state of the sFlow agent

Procedure

To display the system-wide state of the sFlow agent, including any sFlow parameters, collector configuration, and general statistics, use the **info from state** command in candidate or running mode, or the **info** command in state mode.

Example: Info from state command

```
# info from state with-context system sflow
    system {
        sflow {
            admin-state enable
            sample-rate 1000
            sample-size 256
            ipv6-udp-checksum zero
            collector 1 {
                collector-address 10.1.1.24
                network-instance default
                source-address 10.0.0.1
                port 6343
                next-hop 172.24.71.65
            }
            statistics {
                total-samples-taken 5457
                total-sent-packets 26800
            }
        }
    }
```

8.12 Displaying the status of the sFlow agent

Procedure

Use the **show system sflow status** command in show mode to display the general status of the sFlow agent:

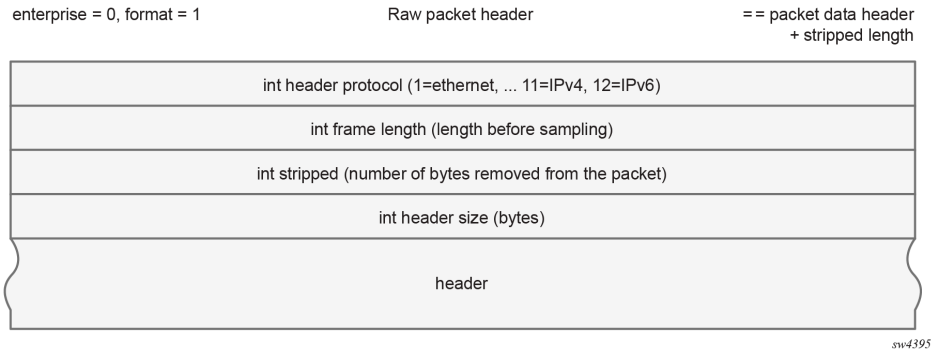
Example: Show system sflow status command

```
--{ running }--[  ]--
# enter show
# show system sflow status
-------------------------------------
Admin State            : enable
Sample Rate            : 10000
Sample Size            : 256
DSCP                   : 0
Total Samples          : 0
Total Collector Packets: 3269158
-------------------------------------
collector-id     : 8
collector-address: 172.10.10.10
network-instance : default
source-address   : 10.0.0.1
port             : 6343
next-hop         : 172.24.71.65
-------------------------------------
```

8.13 sFlow formats

The following figure shows an example of a raw packet header for an sFlow format.

Figure 14: Raw packet header



8.14 Sampled data and counter examples

The following is an example of IPv4 flow sample data:

Example: IPv4 flow sample data

```
InMon sFlow
    Datagram version: 5
    Agent address type: IPv4 (1)
    Agent address: 10.0.0.1
    Sub-agent ID: 2
    Sequence number: 0
    SysUptime: 0
    NumSamples: 1
    Flow sample, seq 0
        0000 0000 0000 0000 0000 .... .... .... = Enterprise: standard sFlow (0)
        .... .... .... .... .... 0000 0000 0001 = sFlow sample type: Flow sample (1)
        Sample length (byte): 141
        Sequence number: 0
        0000 0000 .... .... .... .... .... .... = Source ID class: 0
        .... .... 0000 0000 0000 0000 0011 0110 = Index: 54
        Sampling rate: 1 out of 5 packets
        Sample pool: 0 total packets
        Dropped packets: 0
        Input interface (ifIndex): 54
        .000 0000 0000 0000 0000 0000 0011 0110 = Output interface (ifIndex): 54
        Flow record: 1
        Raw packet header
            0000 0000 0000 0000 0000 .... .... .... = Enterprise: standard sFlow (0)
            Format: Raw packet header (1)
            Flow data length (byte): 101
            Header protocol: Ethernet (1)
            Frame Length: 98
            Payload removed: 0
            Original packet length: 85
            Header of sampled packet:
                000c0002000000000011111108004500005200000000004006...
                Ethernet II, Src: 00:00:00_11:11:11 (00:00:00:11:11:11),
                    Dst: BebIndus_02:00:00 (00:0c:00:02:00:00)
                    Destination: BebIndus_02:00:00 (00:0c:00:02:00:00)
                    Source: 00:00:00_11:11:11 (00:00:00:11:11:11)
                    Type: IPv4 (0x0800)
                Internet Protocol Version 4, Src: 10.100.1.2, Dst: 10.1.1.2
                    0100 .... = Version: 4
                    .... 0101 = Header Length: 20 bytes (5)
                    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
                    Total Length: 82
                    Identification: 0x0000 (0)
                    Flags: 0x00
                    Fragment offset: 0
                    Time to live: 64
                    Protocol: TCP (6)
                    Header checksum: 0x35a1 [validation disabled]
                    [Header checksum status: Unverified]
                    Source: 10.100.1.2
                    Destination: 10.1.1.254
                    [Source GeoIP: Unknown]
                    [Destination GeoIP: Unknown]
                Transmission Control Protocol, Src Port: 0, Dst Port: 0, Seq: 0
                LBT-TCP Protocol
                LBMC Protocol
                [Unreassembled Packet: LBT-TCP]
```

The following is an example of IPv6 flow sample data:

Example: IPv6 flow sample data

```
InMon sFlow
    Datagram version: 5
    Agent address: 3000::2 (3000::2)
    Sub-agent ID: 24
    Sequence number: 1011
    SysUptime: 63684188
    NumSamples: 1
    Flow sample, seq 2368
        Enterprise: standard sFlow (0)
        sFlow sample type: Flow sample (1)
        Sample length (byte): 568
        Sequence number: 2368
        Source ID class: 0 index: 704510
        Sampling rate: 1 out of 1 packets
        Sample pool: 0 total packets
        Dropped packets: 0
        Input interface: ifIndex 134922238
        Output interface: ifIndex 0
        Flow record: 1
        Raw packet header
            Enterprise: standard sFlow (0)
            Format: Raw packet header (1)
            Flow data length (byte): 528
            Header protocol: Ethernet (1)
            Frame Length: 125 bytes
            Payload removed: 0 bytes
            Header of sampled packet: 01005e000002000103ff02018100c064080045c0006b3005...
                Ethernet II, Src: 3com_ff:02:01 (00:01:03:ff:02:01), Dst:
                IPv4mcast_00:00:02 (01:00:5e:00:00:02)
                802.1Q Virtual LAN, PRI: 6, CFI: 0, ID: 100
                Internet Protocol Version 4, Src: 192.35.1.1 (192.35.1.1),
                Dst: 224.0.0.2 (224.0.0.2)
                User Datagram Protocol, Src Port: ldp (646), Dst Port: ldp (646)
                Label Distribution Protocol
                    Version: 1
                    PDU Length: 75
                    LSR ID: 3.3.3.1 (3.3.3.1)
                    Label Space ID: 0
                    Hello Message
```

The following is a counter sample example:

Example: Counters sample

```
InMon sFlow
    Datagram version: 5
    Agent address: 10.0.0.1 (10.0.0.1)
    Sub-agent ID: 0
    Sequence number: 8
    SysUptime: 6548000
    NumSamples: 1
    Counters sample, seq 1
        Enterprise: standard sFlow (0)
        sFlow sample type: Counters sample (2)
        Sample length (byte): 108
        Sequence number: 1
        Source ID type: 64
        Source ID index: 49150
        Counters records: 1
        Generic interface counters
            Enterprise: standard sFlow (0)
```

```
            Format: Generic interface counters (1)
            Flow data length (byte): 88
            Interface index: 1073790974
            Interface Type: 6
            Interface Speed: 25600
            IfDirection: Full-Duplex
            IfAdminStatus: Up
            IfOperStatus: Up
            Input Octets: 0
            Input Packets: 0
            Input Multicast Packets: 0
            Input Broadcast Packets: 0
            Input Discarded Packets: 0
            Input Errors: 0
            Input Unknown Protocol
        Packets: 0
            Output Octets: 0
            Output Packets: 0
            Output Multicast Packets: 0
            Output Broadcast Packets: 0
            Output Discarded Packets: 0
            Output Errors: 0
            Promiscuous Mode: 0
```

9 IPFIX

Note: This feature is supported on 7730 SXR platforms.

IP Flow Information Export (IPFIX) is a tool used to sample IPv4, IPv6, MPLS, and Ethernet traffic data flows through a router. IPFIX enables traffic sampling and analysis by network users and network engineers to support capacity planning, trend analysis, and workload characterization in a network service provider environment.

IPFIX is also known as cflowd version 10 which is an IETF-standardized protocol used to export detailed network flow information from devices to external collectors. IPFIX provides flexible, extensible, and vendor-neutral visibility into network traffic for monitoring, analytics, security, and accounting use cases.

IPFIX is defined by RFC 7011–7015 and represents the standardized evolution of NetFlow v9.

Key benefits

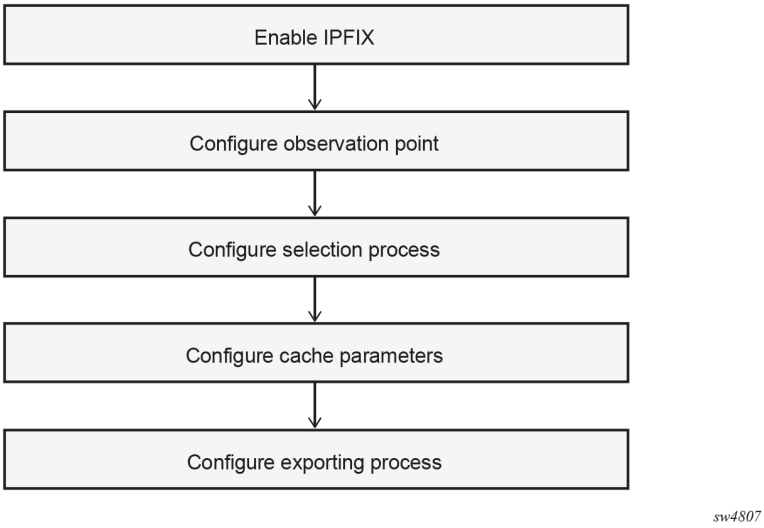
The following are the key benefits of IPFIX:

- Standards-based and vendor-neutral.
- Highly extensible through templates and custom fields.
- Supports modern networking technologies (IPv6, MPLS, overlays).
- Scales to high-speed and large-scale deployments.
- Enables deep traffic visibility for operations and security teams.

9.1 IPFIX operation

The following figure shows the basic operation of IPFIX. This example only describes the basic IPFIX operation overview and is not intended to specify implementation.

Figure 15: IPFIX operation



Observation point and domain

The workflow begins at the observation point. An observation point is the exact location in the forwarding path where packets are examined for potential inclusion in flow processing. Every observation point belongs to a single observation domain. The observation domain provides logical separation for export streams and is identified by a unique 32-bit observation domain ID. This identifier is included in every IPFIX message sent to the collector. Its purpose is to allow collectors to distinguish between different logical metering contexts, particularly in systems with multiple line cards, multiple routing instances, or distributed processing architectures.

Selection process

Not every packet must participate in flow accounting. Before any flow state is created, the packet is evaluated by the Selection process. Each selection process is identified by a unique name, allowing multiple processes to operate concurrently with independent configurations. SR Linux supports sample-count-based sampling where the packets are selected at regular intervals, defined by configurable parameters such as packet interval, the frequency at which packets are sampled and optional packet spacing, memory, or spacing considerations for sampled packets.

Flow cache operation

After a packet is admitted for processing, the metering process extracts the relevant header fields needed to determine flow membership. A flow is defined as a set of packets sharing identical flow keys during a specific time interval. These flow keys typically include source and destination IP addresses, transport ports, protocol number, ingress interface, and potentially additional attributes such as VLAN ID, MPLS labels, or BGP next-hop information.

The metering process performs a lookup in the flow cache to determine whether a matching flow already exists. If no matching flow is found, a new flow record is created. This record contains the flow keys, an initial packet count of one, the corresponding byte count, and the timestamp marking the beginning of the flow. If a matching flow exists, the metering process updates the existing flow record. The packet counter

and byte counter are incremented, and the last-seen timestamp is refreshed. This classification and update process occurs for every selected packet.

The flow cache is a temporary state database that holds all active flows. It allows the system to aggregate multiple packets into a single Flow Record instead of exporting individual packet events. Each flow record maintains both identity and statistical information. In addition to counters and timestamps, the record may include metadata such as TCP flags observed during the flow, forwarding status, autonomous system information, or MPLS label stack data. Flows remain in the cache until one of the expiration conditions occurs:

- An active timeout forces export of long-lived flows, even if traffic is ongoing. This prevents flows from remaining in the cache indefinitely and ensures periodic reporting for persistent sessions.
- An idle timeout expires a flow when no new packets have been observed for a configured duration. This mechanism ensures that completed conversations are exported promptly.
- The system may evict older or least-recently-used flows to make room for new entries. When eviction occurs, the flow is exported before removal to preserve accounting integrity.

When a flow expires for any reason, it transitions from the metering process to the exporting process.

Template formatting and record preparation

Before a flow record can be transmitted to a collector, it must be formatted according to an IPFIX template. IPFIX uses a template-based export model. A template defines the structure of the data record, specifying exactly which information elements are included, their order, and their length. This design allows the exporter to support variable-length records and to adapt the record structure depending on traffic type. Templates are transmitted to the collector in advance of any data records. The collector stores the template and uses it to decode subsequent data sets.

When a flow expires, the exporting process encodes the flow record according to the appropriate template. Each field is serialized into a data record, which is then added to a data set associated with that template ID. If multiple expired flows share the same structure, they are grouped into the same data set for efficiency.

IPFIX message

After data records are prepared, the system constructs an IPFIX message. The message begins with a fixed header containing the protocol version (10), the total message length, the export timestamp, a sequence number, and the observation domain ID. The sequence number increments with every data record exported. This enables collectors to detect message loss when using unreliable transport protocols such as UDP. Following the header, one or more sets are appended. These sets may contain templates, options templates, or data records. Templates are periodically retransmitted to ensure that collectors remain synchronized, particularly in long-lived sessions or after collector restarts. After the message reaches the configured size threshold or timer threshold, it is transmitted to the collector.

Exporting process

The IPFIX messages are transmitted to the collector over UDP using the configured source and destination IP addresses and port. The maximum packet size and template refresh interval are configurable to ensure reliable delivery and correct interpretation of flow data.

In addition to standard flow records, the exporter can send options templates, which provide supplementary metadata such as interface identifiers, exporter statistics, or other context. Options templates are transmitted at a configured interval to maintain the collector’s awareness of the exporting device’s state.

9.2 Configuring IPFIX

About this task

Perform the following steps to configure IPFIX:

Procedure

Step 1. Configure the interfaces as shown in the example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context interface ethernet-1/1
    interface ethernet-1/1 {
        description toward_ixia_1_10_11
        admin-state enable
        vlan-tagging true
        subinterface 1 {
            description to_ixia
            admin-state enable
            ipv4 {
                admin-state enable
                address 30.0.0.1/16 {
                }
            }
            vlan {
                encap {
                    single-tagged {
                        vlan-id 1
                    }
                }
            }
        }
    }
```

```
--{ + candidate shared default }--[  ]--
# info with-context interface ethernet-1/32
    interface ethernet-1/32 {
        description toward_ixia_1_10_16
        admin-state enable
        vlan-tagging true
        subinterface 1 {
            admin-state enable
            ipv4 {
                admin-state enable
                address 40.0.0.1/16 {
                }
            }
            vlan {
                encap {
                    single-tagged {
                        vlan-id 1
                    }
                }
            }
        }
    }
```

Step 2. Configure the network instances as shown in the example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default
    network-instance default {
        type ip-vrf
        admin-state enable
        interface ethernet-1/1.1 {
            interface-ref {
                interface ethernet-1/1
                subinterface 1
            }
        }
    }
```

```
--{ + candidate shared default }--[  ]--
# info with-context network-instance default
    network-instance default {
        type ip-vrf
        admin-state enable
        interface ethernet-1/32.1 {
            interface-ref {
                interface ethernet-1/32
                subinterface 1
            }
        }
    }
```

Step 3. Configure the exporting process as shown in the example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context system ipfix exporting-process exp-1
    system {
        ipfix {
            exporting-process exp-1 {
                export-mode parallel
                destination dest-2 {
                    udp-exporter {
                        ipfix-version 10
                        destination-ip-address 30.0.0.2
                        network-instance default
                    }
                }
            }
        }
    }
```

Step 4. Configure cache parameters as shown in the example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context system ipfix cache cache-1
    system {
        ipfix {
            cache cache-1 {
                exporting-process [
                    exp-1
                ]
```

```
                timeout-cache {
                    maximum-flows 500000
                    active-timeout 180
                    idle-timeout 70
                }
            }
        }
    }
```

Step 5. Configure the selection process as shown in the example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context system ipfix selection-process sel-proc-1
    system {
        ipfix {
            selection-process sel-proc-1 {
                cache cache-1
                selector selector-1 {
                    sample-count-based {
                        packet-interval 1
                        packet-space 1
                    }
                }
            }
        }
    }
```

Step 6. Configure the observation point as shown in the example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context system ipfix observation-point obs-point-1
    system {
        ipfix {
            observation-point obs-point-1 {
                observation-domain-id 11111
                selection-process [
                    sel-proc-1
                ]
                interface ethernet-1/1.1 {
                    direction input
                }
            }
        }
    }
```

Step 7. Enable IPFIX as shown in the example.

Example

```
--{ + candidate shared default }--[  ]--
# info with-context system ipfix admin-state
    system {
        ipfix {
            admin-state enable
        }
    }
```

9.3 IPFIX show commands

Procedure

Use the following show commands to display IPFIX status, collector, and interface information.

Example: Show IPFIX status

```
--{ + candidate shared default }--[  ]--
# show system ipfix status
=============================================================================
IPFIX Admin Status: enable
IPFIX Oper Status : up
=============================================================================
selection process: sel-proc-1
selector         : selector-1
Cache            : cache-1
Packet Space     : 1
Packets Observed : 489026
Packets Dropped  : 112333
=============================================================================
cache name          : cache-1
Active Timeout      : 180
Idle Timeout        : 70
Cache Size          : 500000
Active Flows        : 5
=============================================================================
```

Example: Show IPFIX exporting process detail

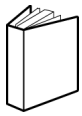
```
--{ + candidate shared default }--[  ]--
# show system ipfix exporting-process destination dest-2 detail
=============================================================================
exporting process       : exp-1
destination name        : dest-2
Host Address            : 30.0.0.2
Port                    : 4739
Version                 : 10
template refresh timeout: 600
network instance        : default
Pkts Sent               : 9450
Last Changed            : 2026-02-24T16:01:49.450Z
=============================================================================
```

Example: Show IPFIX observation point information

```
--{ + candidate shared default }--[  ]--
# show system ipfix observation-point interface ethernet-1/1.1
=============================================================================
Observation Point     : obs-point-1
Observation Domain ID : 11111

-----------------------------------------------------------------------------
Interface name : ethernet-1/1.1
Direction      : input
=============================================================================
```

Customer document and product support



Customer documentation

Customer documentation welcome page

Technical support

Product support portal

Documentation feedback

Customer documentation feedback