Configuring Traffic Mirroring

This module describes the configuration of the traffic mirroring feature. Traffic mirroring is sometimes called port mirroring, or switched port analyzer (SPAN). You can then pass this traffic to a destination port on the same router.

Feature Release History

Release	Modification
Release 6.1.3	ERSPAN Traffic to a Destination Tunnel in a Default VRF was introduced.
Release 7.0.2	SPAN over Pseudo-Wire was introduced.
Release 7.1.2	SPAN to File was introduced.
Release 7.3.1	PCAPng file format was introduced.
Release 7.5.2	Mirror first option in global configuration mode was introduced.
Release 7.5.3	ERSPAN Traffic to a Destination Tunnel in a Non-Default VRF was introduced.
Release 7.6.1	VLAN Sub-interface as Ingress or Egress Source for Traffic Mirroring was introduced.
Release 7.11.1	Traffic Mirroring of Incoming and Outgoing Traffic Separately over Pseudowire was introduced.

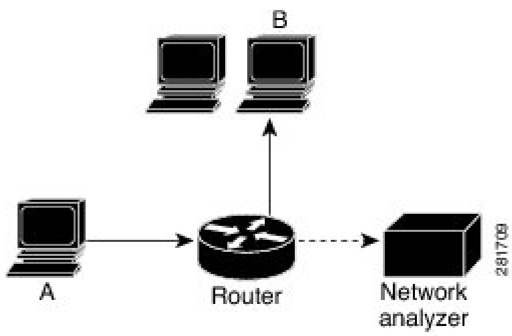
Introduction to Traffic Mirroring

Traffic mirroring, also referred to as Port mirroring or Switched Port Analyzer (SPAN), is a Cisco proprietary feature that enables you to monitor network traffic passing in or out of a set of ports on a router. You can then mirror this traffic to a remote destination or a destination port on the same router.

Traffic mirroring copies traffic from one or more source ports and sends the copied traffic to one or more destinations for analysis by a network analyzer or other monitoring devices. Traffic mirroring does not affect the flow of traffic on the source interfaces or sub-interfaces. It allows the mirrored traffic to be sent to a destination interface or sub-interface.

For example, you can attach a traffic or network analyzer to the router and capture the ethernet traffic that is sent by host A to host B.

Figure 1: Traffic Mirroring Operation



Traffic Mirroring Terminology

- Ingress Traffic — Traffic that comes into the router.
- Egress Traffic — Traffic that goes out of the router.
- Source port—A port that is monitored with the use of traffic mirroring. It is also called a monitored port.
- Destination port—A port that monitors source ports, usually where a network analyzer is connected. It is also called a monitoring port.
- Monitor session—A designation for a collection of SPAN configurations consisting of a single destination and, potentially, one or many source ports.

Traffic Mirroring Types

These are the supported traffic mirroring types.

- Local SPAN
- Remote SPAN
- SPAN on Layer 2 Interfaces
- ACL-based SPAN
- ERSPAN
- SPAN over Pseudo-Wire
- SPAN-to-File, on page 29
- File Mirroring

Characteristics of Source Port

A source port, also called a monitored port, is a routed port that you monitor for network traffic analysis. In a single traffic mirroring session, you can monitor source port traffic. The routers support a maximum of up to 800 source ports.

A source port has these characteristics:

- It can be any data port type, such as Bundle Interface, 100 Gigabit Ethernet physical port, or 10 Gigabit Ethernet physical port.
- Each source port can be monitored in only one traffic mirroring session.
- When a port is used as a source port, the same port cannot be used as a destination port.
- Each source port can be configured with a direction (ingress, egress, or both) to monitor local traffic mirroring. Remote traffic mirroring is supported both in the ingress and egress directions. For bundles, the monitored direction applies to all physical ports in the group.

Characteristics of Monitor Session

A monitor session is a collection of traffic mirroring configurations consisting of a single destination and, potentially, many source interfaces. For any given monitor session, the traffic from the source interfaces (called *source ports*) is sent to the monitoring port or destination port. If there are more than one source port in a monitoring session, the traffic from the several mirrored traffic streams is combined at the destination port. The result is that the traffic that comes out of the destination port is a combination of the traffic from one or more source ports.

Monitor sessions have these characteristics:

- A single monitor session can have only one destination port.
- A single destination port can belong to only one monitor session.
- A monitor session can have a maximum of 800 source ports. This maximum limit is applicable only when the maximum number of source ports from all monitoring sessions does not exceed 800.

Characteristics of Destination Port

Each session must have a destination port or file that receives a copy of the traffic from the source ports.

A destination port has these characteristics:

- A destination port cannot be a source port.
- For local traffic mirroring, a destination port must reside on the same router as the source port.
- For remote mirroring, the destination is always a GRE tunnel.
- A destination port for local mirroring can be any Ethernet physical port, EFP, GRE tunnel interface, or bundle interface. It can be a Layer 2 or Layer 3 transport interface.
- A destination port on router cannot be a VLAN subinterface.

- At any time, a destination port can participate in only one traffic mirroring session. A destination port in one traffic mirroring session cannot be a destination port for a second traffic mirroring session. In other words, no two monitor sessions can have the same destination port.

Supported Scale

This list provides scale supported on the NCS 560 routers for traffic mirroring.

- Prior to Cisco IOS XR Release 7.8.1, a single router could support up to four monitor sessions. However, configuring SPAN and CFM on the router reduced the maximum number of monitor sessions to two, as both shared the mirror profiles.
- Starting Cisco IOS XR Software Release 7.8.1, SPAN supports a maximum of up to three monitor sessions on the NCS 560 routers. But, if you configure SPAN and CFM on the router, the maximum number of monitor sessions decreases to one, as both functions use the same mirror profiles. The decrease in the number of monitor sessions does not affect the NCS 5700 platforms.

Restrictions

Generic Restrictions

The following are the generic restrictions related to traffic mirroring:

- Partial mirroring and sampled mirroring are not supported.
- From Release 7.6.1, sub-interface configured as source interface is supported on SPAN.
- The destination bundle interfaces flap when:
  - both the mirror source and destination are bundle interfaces in the Link Aggregation Control Protocol (LACP) mode.
  - mirror packets next-hop is a router or a switch instead of a traffic analyzer.

  This behavior is observed due to a mismatch of LACP packets on the next-hop bundle interface due to the mirroring of LACP packets on the source bundle interface.
- Subinterface with only one VLAN is supported as source for traffic mirroring.
- Bridge group virtual interfaces (BVIs) are not supported as source ports or destination ports.
- Bundle members cannot be used as destination ports.
- Fragmentation of mirror copies is not handled by SPAN when SPAN destination MTU is less than the packet size. Existing behaviour if the MTU of destination interface is less than the packet size is as below:

Platforms	Rx SPAN	Tx SPAN
NCS 560	Mirror copies are not fragmented. Receives whole packets as mirror copies.	Mirror copies are fragmented.

  You can configure the SPAN destination with an MTU which is greater than the packet size.

- Until Cisco IOS XR Software Release 7.6.1, SPAN only supports port-level source interfaces.
- Packets arriving at the subinterface will not be mirrored if Rx SPAN is enabled on the main bundle interface.

Restrictions on VLAN Sub-interface as Source

- Supports a maximum of 24 reception and transmission sessions together for mirroring. This restriction is applicable for sub-intefaces and ports as source.
- When the port is in Egress Traffic Management (ETM) mode, the outgoing or egress (Tx) traffic mirroring is possible only on the sub-interface for which the egress (Tx) traffic mirroring is configured.
- Tx mirroring is applicable on ETM mode only. Rx mirroring is applicable on both the ETM and non-ETM modes.

Restrictions on ACL-based SPAN

The following restrictions apply to SPAN-ACL:

Table 1: SPAN-ACL Support

Platforms	Rx Direction	Tx Direction
NCS 540	Supported at the port level, that is, in the ingress direction for IPv4 or IPv6 ACLs.	Not supported.

- MPLS traffic cannot be captured with SPAN-ACL.
  - ACL for any MPLS traffic is not supported.
- Traffic mirroring counters are not supported.
- ACL-based traffic mirroring is not supported with Layer 2 (ethernet-services) ACLs.
- Main interface as span source interface and ACL with the **capture** keyword on same main interface's sub-interface are not supported.
- If a SPAN session with the **acl** keyword is applied on an interface with no ACL rule attached to that interface, SPAN happens without any filtering.

Restrictions on ERSPAN

This section provides the restrictions that apply to ERSPAN and multiple ERSPAN sessions.

The following restrictions apply to ERSPAN:

- ERSPAN next-hop must have ARP resolved.
- ERSPAN packets with outgoing interface having MPLS encapsulation are not supported. The next-hop router or any router in the path can encapsulate in MPLS.
  - Additional routers may encapsulate in MPLS.

- ERSPAN sessions can be created only on physical interfaces. The sessions cannot be created on sub-interfaces.
- ERSPAN supports a maximum of three sessions.
- ERSPAN decapsulation is not supported.
- ERSPAN does not work if the GRE next hop is reachable over sub-interface. For ERSPAN to work, the next hop must be reachable over the main interface.
- ERSPAN decapsulation is not supported. Tunnel destination should be network analyzer.
- ERSPAN is not supported when the **hw-module profile segment-routing srv6 mode micro-segment format f3216** configuration is enabled.

Restrictions on SPAN over Pseudowire

SPAN over Psedowire (PW-SPAN) has the following restrictions:

- PW-SPAN does not support the listed functionalities:
  - Monitor session statistics
  - Partial packet SPAN
  - Sampled SPAN
- ETM mode must be enabled for outgoing (Tx) traffic on sub-interface.

Restrictions on SPAN-to-File

SPAN to File has the following restrictions:

- A maximum of 1000 source ports are supported across the system. Individual platforms may support lower numbers. The SPAN session may be any of these currently supported classes: Ethernet, IPv4, IPv6, MPLS-IPv4, and MPLS-IPv6.
- Provides a buffer range of 1000-1000000 KB. The default buffer size is set to 1000 KB.
- Provides support for SPAN source.
  - Each source port can be monitored in only one traffic mirroring session.
  - Each source port can be configured with a direction (ingress, egress, or both) to monitor local traffic mirroring.
- Only port-level is supported.
- VLAN interface as source port is not supported.
- Bundle members as source interfaces are not supported.
- Filtering based on Egress ACL is not supported.
- Source port statistics is not supported.
- Span to file mirror packets are punted from NPU to CPU at a maximum shaper rate of 40 mbps.

Restrictions on File Mirroring

The following restrictions apply to file mirroring:

- Supported only on Dual RP systems.
- Supports syncing only from active to standby RP. If files are copied into standby `/harddisk:/mirror` location, it won't be synced to active RP.
- A slight delay is observed in `show mirror` command output when mirror checksum configuration is enabled.
- Not supported on multichassis systems.

Restrictions on Forward-Drop Packets Mirroring

These are some restrictions for Forward-Drop packets mirroring:

- Only one global forward-drop session can be configured on a router.
- When traffic-class is configured under monitor-session for forward-drop, the type of service (ToS) byte of the outgoing ERSPAN packet is overwritten with the configured traffic-class value.
- In-band traffic destined to router management interface cannot be captured using this functionality.
- Forward-drop packets mirroring does not support access control lists (ACL) drops.

SPAN Types, Supported Features, and Configurations

Local SPAN

This is the most basic form of traffic mirroring. The network analyzer or sniffer is attached directly to the destination interface. In other words, all monitored ports are located on the same router as the destination port.

Remote SPAN

Configure Remote Traffic Mirroring

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **monitor-session** *session-name*

Example:

```
RP/0/RP0/CPU0:router(config)# monitor-session mon1 ethernet
RP/0/RP0/CPU0:router(config-mon)#
```

Defines a monitor session and enters monitor session configuration mode.

Step 3 **destination interface** *subinterface*

Example:

```
RP/0/RP0/CPU0:router(config-mon)# destination interface TenGigE 0/2/0/4.1
```

Specifies the destination subinterface to which traffic is replicated.

Step 4 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-mon)# exit
RP/0/RP0/CPU0:router(config)#
```

Exits monitor session configuration mode and returns to global configuration mode.

Step 5 **interface** *type number*

Example:

```
RP/0/RP0/CPU0:router(config)# interface HundredGigE 0/0/1/0
```

Enters interface configuration mode for the specified source interface. The interface number is entered in *rack*/*slot*/*module*/*port* notation. For more information about the syntax for the router, use the question mark (?) online help function.

Step 6 **monitor-session** *session-name* **ethernet direction rx-onlyport-only**

Example:

```
RP/0/RP0/CPU0:router(config-if)# monitor-session mon1 ethernet
direction rx-only port-only
```

Specifies the monitor session to be used on this interface. Use the **direction** keyword to specify that only ingress or egress traffic is mirrored.

Step 7 **end** or **commit**

Example:

```
RP/0/RP0/CPU0:router(config-if)# end
```

or

```
RP/0/RP0/CPU0:router(config-if)# commit
```

Saves configuration changes.

- When you issue the **end** command, the system prompts you to commit changes:

```
Uncommitted changes found, commit them before exiting (yes/no/cancel)?
[cancel]:
```

- Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.

- Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.

- Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 8 **show monitor-session [session-name]** **status** [**detail**] [**error**]

Example:

```
RP/0/RP0/CPU0:router# show monitor-session
```

Displays information about the traffic mirroring session.

Example

This example shows the basic configuration for traffic mirroring with physical interfaces.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# monitor-session ms1
RP/0/RP0/CPU0:router(config-mon)# destination interface HundredGigE0/2/0/15
RP/0/RP0/CPU0:router(config-mon)# commit

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE0/2/0/19
RP/0/RP0/CPU0:router(config-if)# monitor-session ms1 port-level
RP/0/RP0/CPU0:router(config-if)# commit

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE0/2/0/19
RP/0/RP0/CPU0:router(config-if)# monitor-session ms1 direction rx-only port-level
RP/0/RP0/CPU0:router(config-if)# commit

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE0/2/0/19
RP/0/RP0/CPU0:router(config-if)# monitor-session ms1 direction tx-only port-level
RP/0/RP0/CPU0:router(config-if)# commit
```

This example shows sample output of the show monitor-session command with the status keyword:

```
RP/0/RSP0/CPU0:router# show monitor-session status
Monitor-session cisco-rtp1
Destination interface HundredGigE 0/5/0/38
================================================================================
Source Interface Dir Status
--------------------- ---- ----------------------------------------------------
TenGigE0/5/0/4 Both Operational
TenGigE0/5/0/17 Both Operational
RP/0/RSP0/CPU0:router# show monitor-session status detail
Monitor-session sess1
Destination interface is not configured
```

```
Source Interfaces
-----------------
TenGigE0/2/0/19
Direction: Both
ACL match: Disabled
Portion: Full packet
Status: Not operational (destination interface not known).
TenGigE0/1/0/1
Direction: Both
ACL match: Disabled
Portion: First 100 bytes

RP/0/RSP0/CPU0:router# show monitor-session status error
Monitor-session ms1
Destination interface TenGigE0/2/0/15 is not configured
================================================================================
Source Interface Dir Status
--------------------- ---- ----------------------------------------------------
Monitor-session ms2
Destination interface is not configured
================================================================================
Source Interface Dir Status
--------------------- ---- ----------------------------------------------------
RP/0/RP0/CPU0:router# show monitor-session test status
Monitor-session test (ipv4)
Destination Nexthop 255.254.254.4
=========================================================================================
Source Interface Dir Status
-----------------------------------------------------------------------------------------
Gi0/0/0/2.2 Rx Not operational (source same as destination)
Gi0/0/0/2.3 Rx Not operational (Destination not active)
Gi0/0/0/2.4 Rx Operational
Gi0/0/0/4 Rx Error: see detailed output for explanation
RP/0/RP0/CPU0:router# show monitor-session test status error
Monitor-session test
Destination Nexthop ipv4 address 255.254.254.4
================================================================
Source Interface Status
----------------------------------------------------------------
Gi0/0/0/4 < Error: FULL Error Details >
```

SPAN on Subinterfaces

SPAN can be configured on up to six subinterfaces (either physical subinterfaces or bundle subinterfaces) associated with a single physical interface.

VLAN Subinterface as Ingress or Egress Source for Traffic Mirroring

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
VLAN Subinterface as Ingress or Egress Source for Traffic Mirroring	Release 7.6.1	You can now configure the VLAN subinterface as an egress or ingress source for traffic mirroring. This feature enables the monitoring of traffic mirrored on either egress or ingress or both directions. You could configure mirror functionality only at the main interface level in earlier releases.

VLAN subinterface provides the flexibility to monitor ingress or egress, or both ingress/egress traffic from all the active subinterfaces of the source VLAN. The active subinterfaces in the source VLAN are considered as source subinterfaces. When subinterfaces are added or removed from the source VLAN, the corresponding traffic is added or removed from the monitoring sources.

VLAN Subinterface as Ingress Source for Traffic Mirroring

Configuration Example

```
Router# configure
Router(config)# monitor-session mon1 ethernet
Router(config-mon)# destination interface tunnel-ip 3
Router(config-mon)# exit
Router(config)# interface HundredGigE 0/1/0/1.10
Router(config-subif)#
Router(config-if-mon)# commit
```

Running Configuration

```
Router# show run monitor-session mon1
monitor-session mon1 ethernet
 destination interface tunnel-ip3
!
```

```
Router# show run interface HundredGigE 0/1/0/1.10
 interface HundredGigE0/1/0/1.10
  encapsulation dot1q 10
   ipv4 address 101.1.2.1 255.255.255.252
    monitor-session mon1 ethernet
    !
   !
  !
```

Verification

Verify that the status for VLAN subinterface is in the operational state for the incoming (Rx) traffic by using the **show monitor-session status command**:

```
Router# show monitor-session status
Monitor-session mon1
Destination interface tunnel-ip3
================================================================================
Source Interface Dir Status
--------------------- ---- ----------------------------------------------------
HundredGigE 0/1/0/1.10  Both Operational
```

VLAN Interface as Egress Source for Traffic Mirroring

Configuration Example

Running Configuration

```
Router# show run monitor-session mon1
monitor-session mon1 ethernet
 destination interface tunnel-ip3
!
```

```
Router# show run interface HundredGigE 0/1/0/1.10
interface HundredGigE0/1/0/1.10
 encapsulation dot1q 20
  ipv4 address 102.1.2.1 255.255.255.252
   monitor-session mon1 ethernet
    !
   !
  !
```

Verification

Verify that the status for VLAN subinterface is in the operational state for the outgoing (Tx) traffic by using the **show monitor-session status command**:

```
Router# show monitor-session status
Monitor-session mon1
Destination interface tunnel-ip3
================================================================================
Source Interface Dir Status
--------------------- ---- ----------------------------------------------------
HundredGigE 0/1/0/1.10  Both Operational
```

Monitoring Traffic Mirroring on a Layer 2 Interface

This section describes the configuration for monitoring traffic on a Layer 2 interface.

Configuration

To monitor traffic mirroring on a Layer 2 interface, configure the monitor under `l2transport` sub-config of the interface:

```
RP/0/RP0/CPU0:router(config)# interface TenGigE0/0/0/42
RP/0/RP0/CPU0:router(config-if)# l2transport
RP/0/RP0/CPU0:router(config-if-l2)# monitor-session EASTON ethernet port-level
```

Verification

Verify that the status for traffic mirroring on a Layer 2 interface is in the operational state by using the **show monitor-session status command**:

```
RP/0/RP0/CPU0:router# show monitor-session status
Thu Aug 29 21:42:22.829 UTC
Monitor-session EASTON
Destination interface TenGigE0/0/0/20
=============================================
Source Interface      Dir   Status
--------------------- ----  -----------------
Te0/0/0/42 (port)     Both  Operational
```

ACL-based SPAN

Traffic is mirrored based on the configuration of the interface ACL.

You can mirror traffic based on the definition of an interface access control list. When you mirror Layer 3 traffic, the ACL is configured using the **ipv4 access-list** or the **ipv6 access-list** command with the **capture** option. The **permit** and **deny** commands determine if the packets in the traffic are permitted or denied. The **capture** option designates the packet is to be mirrored to the destination port, and it is supported only on permit type of Access Control Entries (ACEs).

Note

- Prior to Release 6.5.1, ACL-based traffic mirroring required the use of UDK (User-Defined TCAM Key) with the **enable-capture** option so that the **capture** option can be configured in the ACL.
- ACL must be defined before attaching the ACL name to SPAN source interface.

Configuring Security ACLs for Traffic Mirroring

This section describes the configuration for creating security ACLs for traffic mirroring.

In ACL-based traffic mirroring, traffic is mirrored based on the configuration of the interface ACL. You can mirror traffic based on the definition of an interface access control list. When you're mirroring Layer 3 or Layer 2 traffic, the ACL is configured using the **ipv4 access-list** or the **ipv6 access-list** command with the **capture** option. The **permit** and **deny** commands determine the behavior of the regular traffic.

Configure an IPv4 ACL for Traffic Mirroring

Use the following steps to configure ACLs for traffic mirroring.

```
/* Create an IPv4 ACL (TM-ACL) for traffic mirroring */
Router(config)# ipv4 access-list TM-ACL
Router(config-ipv4-acl)# 10 permit udp 10.1.1.0 0.0.0.255 eq 10 any capture
Router(config-ipv4-acl)# 20 permit udp 10.1.1.0 0.0.0.255 eq 20 any
Router(config-ipv4-acl)# exit
Router(config)# commit

/* Validate the configuration */
Router(config)# show run
Thu May 17 11:17:49.968 IST
Building configuration...
!! IOS XR Configuration 0.0.0
!! Last configuration change at Thu May 17 11:17:47 2018 by user
…
ipv4 access-list TM-ACL
 10 permit udp 10.1.1.0 0.0.0.255 eq 10 any capture
 20 permit udp 10.1.1.0 0.0.0.255 eq 20 any
!
…
```

You have successfully configured an IPv4 ACL for traffic mirroring.

Configuring UDF-Based Security ACL for Traffic Mirroring

Before you begin

This section describes the configuration steps for UDF-based security ACLs for traffic mirroring.

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **udf** *udf-name* **header** {**inner** | **outer**} {**l2** | **l3** | **l4**} **offset** *offset-in-bytes* **length** *length-in-bytes*

Example:

```
RP/0/RP0/CPU0:router(config)# udf udf3 header outer l4 offset 0 length 1
(config-mon)#
```

Example:

```
RP/0/RP0/CPU0:router(config)# udf udf3 header inner l4 offset 10 length 2
(config-mon)#
```

Example:

```
RP/0/RP0/CPU0:router(config)# udf udf3 header outer l4 offset 50 length 1
(config-mon)#
```

Configures individual UDF definitions. You can specify the name of the UDF, the networking header from which offset, and the length of data to be extracted.

The **inner** or **outer** keywords indicate the start of the offset from the unencapsulated Layer 3 or Layer 4 headers, or if there is an encapsulated packet, they indicate the start of offset from the inner L3/L4.

Note
The maximum offset allowed, from the start of any header, is 63 bytes

The **length** keyword specifies, in bytes, the length from the offset. The range is from 1 to 4.

Step 3 **ipv4 access-list** *acl-name*

Example:

```
RP/0/RP0/CPU0:router(config))# ipv4 access-list acl1
```

Creates ACL and enters IP ACL configuration mode. The length of the *acl-name* argument can be up to 64 characters.

Step 4 **permit** *regular-ace-match-criteria* **udf** *udf-name1 value1 ... udf-name8 value8*

Example:

```
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 any any udf udf1 0x1234 0xffff udf3
0x56 0xff capture
RP/0/RP0/CPU0:router(config-ipv4-acl)# 30 permit ipv4 any any dscp af11 udf udf5 0x22 0x22
 capture
```

Configures ACL with UDF match.

Step 5 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-ipv4-acl)# exit
```

Exits IP ACL configuration mode and returns to global configuration mode.

Step 6 **interface***type number*

Example:

```
RP/0/RP0/CPU0:router(config)# interface HundredGigE 0/0/1/0
```

Configures interface and enters interface configuration mode.

Step 7 **ipv4 access-group** *acl-name* **ingress**

Example:

```
RP/0/RP0/CPU0:router(config-if)# ipv4 access-group acl1 ingress
```

Applies access list to an interface.

Step 8 **commit**

Example:

```
RP/0/RP0/CPU0:router(config-if)# commit
```

Applies access list to an interface.

Verifying UDF-based Security ACL

Use the **show monitor-session status detail** command to verify the configuration of UDF on security ACL.

```
RP/0/RP0/CPU0:leaf1# show monitor-session 1 status detail

Fri May 12 19:40:39.429 UTC
Monitor-session 1
  Destination interface tunnel-ip3
  Source Interfaces
  -----------------
  TenGigE0/0/0/15
    Direction:  Rx-only
    Port level: True
    ACL match:  Enabled
    Portion:    Full packet
    Interval:   Mirror all packets
    Status:     Not operational (destination not active)
```

Attaching the Configurable Source Interface

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **interface** *type number*

Example:

```
RP/0/RP0/CPU0:router(config)# interface HundredGigE 0/0/1/0
```

Enters interface configuration mode for the specified source interface. The interface number is entered in *rack*/*slot*/*module*/*port* notation. For more information about the syntax for the router, use the question mark (?) online help function.

Step 3 **ipv4 access-group** *acl-name* {**ingress** | **egress**}

Example:

```
RP/0/RP0/CPU0:router(config-if)# ipv4 access-group acl1 ingress
```

Controls access to an interface.

Step 4 **monitor-session** *session-name* **ethernet direction rx-onlyport-level acl**

Example:

```
RP/0/RP0/CPU0:router(config-if)# monitor-session mon1 ethernet direction rx-only port-level
 acl
RP/0/RP0/CPU0:router(config-if-mon)#
```

Attaches a monitor session to the source interface and enters monitor session configuration mode.

Note
rx-only specifies that only ingress traffic is replicated.

Step 5 **acl**

Example:

```
RP/0/RP0/CPU0:router(config-if-mon)# acl
```

Specifies that the traffic mirrored is according to the defined ACL.

Note
If an ACL is configured by name, then this step overrides any ACL that may be configured on the interface.

Step 6 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-if-mon)# exit
RP/0/RP0/CPU0:router(config-if)#
```

Exits monitor session configuration mode and returns to interface configuration mode.

Step 7 **end** or **commit**

Example:

```
RP/0/RP0/CPU0:router(config-if)# end
```

or

```
RP/0/RP0/CPU0:router(config-if)# commit
```

Saves configuration changes.

- When you issue the **end** command, the system prompts you to commit changes:

```
Uncommitted changes found, commit them before exiting (yes/no/cancel)?
[cancel]:
```

  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.

  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.

  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 8 **show monitor-session [session-name]** **status** [**detail**] [**error**]

Example:

```
RP/0/RP0/CPU0:router# show monitor-session status
```

Displays information about the monitor session.

ERSPAN

Encapsulated Remote Switched Port Analyzer (ERSPAN) transports mirrored traffic over an IP network. The traffic is encapsulated at the source router and is transferred across the network. The packet is decapsulated at the destination router and then sent to the destination interface.

Encapsulated Remote SPAN (ERSPAN) enables generic routing encapsulation (GRE) for all captured traffic and allows it to be extended across Layer 3 domains.

ERSPAN involves mirroring traffic through a GRE tunnel to a remote site. For more information on configuring the GRE tunnel that is used as the destination for the monitor sessions, see the chapter *Configuring GRE Tunnels*.

Note A copy of every packet includes the Layer 2 header if the ethernet keyword is configured. As this renders the mirrored packets unroutable, the end point of the GRE tunnel must be the network analyzer.

Introduction to ERSPAN Egress Rate Limit

With ERSPAN egress rate limit feature, you can monitor traffic flow through any IP network. This includes third-party switches and routers.

ERSAPN operates in the following modes:

- ERSPAN Source Session – box where the traffic originates (is SPANned).
- ERSPAN Termination Session or Destination Session – box where the traffic is analyzed.

This feature provides rate limiting of the mirroring traffic or the egress traffic. With rate limiting, you can limit the amount of egress traffic to a specific rate, which prevents the network and remote ERSPAN destination traffic overloading. Be informed, if the egress rate-limit exceeds then the system may cap or drop the monitored traffic.

You can configure the QoS parameters on the traffic monitor session.

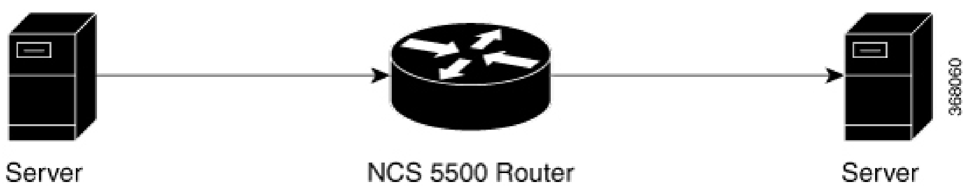
- Traffic Class (0 through 7)
    - Traffic class 0 has the lowest priority and 7 the highest.
    - The default traffic class is the same as that of the original traffic class.
- The Discard Class (0 through 2):
    - The default is 0.
    - The discard class configuration is used in WRED.

Benefits

With ERSPAN Egress rate limit feature, you can limit the egress traffic or the mirrored and use the mirrored traffic for data analysis.

Topology

Figure 2: Topology for ERSPAN Egress Rate Limit



The encapsulated packet for ERSPAN is in ARPA/IP format with GRE encapsulation. The system sends the GRE tunneled packet to the destination box identified by an IP address. At the destination box, SPAN-ASIC decodes this packet and sends out the packets through a port. ERSPAN egress rate limit feature is applied on the router egress interface to rate limit the monitored traffic.

The intermediate switches carrying ERSPAN traffic from source session to termination session can belong to any L3 network.

Configure ERSPAN Egress Rate Limit

Use the following steps to configure ERSPAN egress rate limit:

```
monitor-session ERSPAN ethernet
destination interface tunnel-ip1
!

RP/0/RP0/CPU0:pyke-008#sh run int tunnel-ip 1

interface tunnel-ip1
ipv4 address 4.4.4.1 255.255.255.0
tunnel mode gre ipv4
tunnel source 20.1.1.1
tunnel destination 20.1.1.2
!
```

```
RP/0/RP0/CPU0:pyke-008#sh run int hundredGigE 0/0/0/16

interface HundredGigE0/0/0/16
ipv4 address 215.1.1.1 255.255.255.0
ipv6 address 3001::2/64
monitor-session ERSPAN ethernet direction rx-only port-level
  acl
!
ipv4 access-group ACL6 ingress
```

Running Configuration

```
!! Policy-map to be used with the ERSPAN Destination (egress interface)
!! Traffic class is set to 5. For packets in this class, apply shaping
!! as well as WRED.
class-map match-any TC5
 match traffic-class 5
 end-class-map
!
policy-map shape-foo
 class TC5
  random-detect discard-class 0 10000 bytes 40000 bytes
  random-detect discard-class 1 40000 bytes 80000 bytes
  random-detect discard-class 2 80000 bytes 200000 bytes
  shape average percent 15
 !
 class class-default
 !
 end-policy-map
!
!!GRE Tunnel Interface
interface Loopback49
 ipv4 address 49.49.49.49 255.255.255.255
!
interface tunnel-ip100
 ipv4 address 130.100.1.1 255.255.255.0
 tunnel mode gre ipv4
 tunnel source 49.49.49.49
 tunnel destination 10.8.1.2
!
!!ERSPAN Monitor Session with GRE tunnel as the Destination Interface, and with QoS
configuration
monitor-session FOO ethernet
 destination interface tunnel-ip100
 traffic-class 5
 discard-class 1
!
!!ERSPAN Source Interface
interface TenGigE0/6/0/4/0
 description connected to TGEN 9/5
 ipv4 address 10.4.90.1 255.255.255.0
 monitor-session FOO ethernet port-level
 !
!
!!ERSPAN Destination ip-tunnel00's underlying interface, with egress policy-map shape-foo
attached
interface TenGigE0/6/0/9/0
 service-policy output shape-foo
 ipv4 address 10.8.1.1 255.255.255.0
```

Verification

```
RP/0/RP0/CPU0:ios#show monitor-session FOO status detail
Wed May  2 15:14:05.762 UTC
```

```
Monitor-session FOO
  Destination interface tunnel-ip100
  Source Interfaces
  -----------------
  TenGigE0/6/0/4/0
    Direction:  Both
    Port level: True
    ACL match:  Disabled
    Portion:    Full packet
    Interval:   Mirror all packets
    Status:     Operational
RP/0/RP0/CPU0:ios#
show monitor-session <sess-id> status internal

RP/0/RP0/CPU0:ios#show monitor-session FOO status internal
Wed May  2 15:13:06.063 UTC
Information from SPAN Manager and MA on all nodes:
Monitor-session FOO (ID 0x00000001) (Ethernet)
SPAN Mgr: Destination interface tunnel-ip100 (0x0800001c)
          Last error: Success
          Tunnel data:
            Mode: GREoIPv4
            Source IP: 49.49.49.49
            Dest IP: 10.8.1.2
            VRF:
            ToS: 0 (copied)
            TTL: 255
            DFbit: Not set
0/6/CPU0: Destination interface tunnel-ip100 (0x0800001c)
          Tunnel data:
            Mode: GREoIPv4
            Source IP: 49.49.49.49
            Dest IP: 10.8.1.2
            VRF:
            ToS: 0 (copied)
            TTL: 255
            DFbit: Not set

Information from SPAN EA on all nodes:
Monitor-session 0x00000001 (Ethernet)
0/6/CPU0: Name 'FOO', destination interface tunnel-ip100 (0x0800001c)
Platform, 0/6/CPU0:

  Dest Port: 0xe7d

  ERSPAN Encap:
    Tunnel ID: 0x4001380b
    ERSPAN Tunnel ID: 0x4001380c
    IP-NH Grp key: 0x3140000cc5
    IP-NH hdl: 0x308a5fa5e0
    IP-NH IFH: 0x30002a0
    IP-NH IPAddr: 10.4.91.2

  NPU   MirrorRx     MirrorTx
  00    0x00000003   0x00000004
  01    0x00000003   0x00000004
  02    0x00000003   0x00000004
  03    0x00000003   0x00000004
  04    0x00000003   0x00000004
  05    0x00000003   0x00000004
RP/0/RP0/CPU0:ios#
```

ERSPAN Traffic to a Destination Tunnel in a Default VRF

Table 3: Feature History Table

Feature Name	Release Information	Description
ERSPAN Traffic to a Destination Tunnel in a Default VRF	Release 6.1.3	Encapsulated Remote Switched Port Analyzer (ERSPAN) now transports mirrored traffic through GRE tunnels that belongs to the default VRF thus ensuring a network design with a single Layer 3 device. This feature enables the tunnels to be grouped under the default VRF domain towards which you can segregate the traffic.

Running Configuration

The following example shows a tunnel interface configured with endpoints in a default VRF (**vrf: green**):

```
Router#show run int tunnel-ip 2
Thu Feb  3 06:18:28.075 UTC
interface tunnel-ip2
 ipv4 address 102.1.1.100 255.255.255.0
 tunnel tos 32
 tunnel mode gre ipv4
 tunnel source 120.1.1.100
 tunnel vrf green
 tunnel destination 120.1.1.1
```

```
Router#show monitor-session status
Thu Feb  3 06:18:11.061 UTC
Monitor-session ERSPAN-2
Destination interface tunnel-ip2
===============================================================
Source Interface      Dir   Status
--------------------- ----  ----------------------------------------------------
Te0/0/0/5 (port)      Rx    Operational
```

Verification

The following CLI output shows how to verify the default VRF configuration:

```
Router#show monitor-session ERSPAN-2 status internal
Thu Feb  3 06:19:50.014 UTC

Information from SPAN Manager and MA on all nodes:
Monitor-session ERSPAN-2 (ID 0x00000003) (Ethernet)
SPAN Mgr: Destination interface tunnel-ip2 (0x20008024)
          Last error: Success
          Tunnel data:
            Mode: GREoIPv4
            Source IP: 120.1.1.100
            Dest IP: 120.1.1.1
            VRF: green
            VRF TBL ID: 0
```

```
ToS: 32
TTL: 255
DFbit: Not set
```

ERSPAN Traffic to a Destination Tunnel in a Non-Default VRF

Table 4: Feature History Table

Feature Name	Release Information	Description
ERSPAN Traffic to a Destination Tunnel in a Non-Default VRF	Release 7.5.3	The tunnels are grouped under the VRFs and you can segregate the traffic towards a specific VRF domain. Encapsulated Remote Switched Port Analyzer (ERSPAN) now transports mirrored traffic through GRE tunnels with multiple VRFs, helping you design your network with multiple Layer 3 partitions. In earlier releases, ERSPAN transported mirrored traffic through GRE tunnels that belonged to only default VRF.

Here, the tunnel interface, where the traffic mirroring is destined, is now in a VRF.

The traffic coming out of the interfaces of a router do not have any grouping. By configuring a specific VRF, you can now identify the incoming traffic group.

Configuration

Use the following command to configure a specific VRF:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface tunnel-ip 2
RP/0/RP0/CPU0:router(config)# tunnel vrf red
```

For more information on enabling the tunnel mode in GRE, see Configuring GRE Tunnels.

Configuration example

The following example shows a tunnel interface configured with endpoints in a non-default VRF (**vrf: red**):

```
Router#show run int tunnel-ip 2
Thu Feb  3 06:18:28.075 UTC
interface tunnel-ip2
 ipv4 address 102.1.1.100 255.255.255.0
 tunnel tos 32
 tunnel mode gre ipv4
 tunnel source 120.1.1.100
 tunnel vrf red
 tunnel destination 120.1.1.1
```

```
Router#show monitor-session status
Thu Feb  3 06:18:11.061 UTC
Monitor-session ERSPAN-2
```

```
Destination interface tunnel-ip2
================================================================
Source Interface      Dir   Status
--------------------- ----  ----------------------------------------------------
Te0/0/0/5 (port)      Rx    Operational
```

Verification

The following CLI output shows how to verify, if the configured tunnel VRF is programmed in the session:

```
Router#show monitor-session ERSPAN-2 status internal
Thu Feb  3 06:19:50.014 UTC

Information from SPAN Manager and MA on all nodes:
Monitor-session ERSPAN-2 (ID 0x00000003) (Ethernet)
SPAN Mgr: Destination interface tunnel-ip2 (0x20008024)
          Last error: Success
          Tunnel data:
            Mode: GREoIPv4
            Source IP: 120.1.1.100
            Dest IP: 120.1.1.1
            VRF: red
            VRF TBL ID: 0
            ToS: 32
            TTL: 255
            DFbit: Not set
```

SPAN over Pseudowire

Pseudo-wire traffic mirroring (known as PW-SPAN) is an extra functionality on the existing SPAN solutions. The existing SPAN solutions are monitored on a destination interface or through a GRE tunnel or RSPAN. In PW-SPAN, the traffic mirroring destination port is configured to be a pseudo-wire rather than a physical port. Here, the designated traffic on the source port is mirrored over the pseudo-wire to a central location. This allows the centralization of expensive network traffic analysis tools.

Because the pseudo-wire carries only mirrored traffic, this traffic is unidirectional. Incoming traffic from the remote provider edge is not allowed. Typically, a monitor session should be created with a destination pseudo-wire. This monitor session is one of the L2VPN xconnect segments. The other segment of the L2VPN VPWS is a pseudowire.

Configure SPAN over Pseudowire

Use the following steps to configure SPAN over Pseudowire:

Configure SPAN monitor session

```
RP/0/RP0/CPU0:router#config
RP/0/RP0/CPU0:router(config)#monitor-session M1
RP/0/RP0/CPU0:router(config-mon)#destination pseudowire
RP/0/RP0/CPU0:router(config-mon)#commit
```

Configure SPAN source

```
RP/0/RP0/CPU0:router#config
Fri Sep  6 03:49:59.312 UTC
RP/0/RP0/CPU0:router(config)#interface Bundle-Ether100
RP/0/RP0/CPU0:router(config-if)#monitor-session M1 ethernet port-level
RP/0/RP0/CPU0:router(config-if-mon)#commit
```

Configure l2vpn xconnect

```
RP/0/RP0/CPU0:router(config)#l2vpn
RP/0/RP0/CPU0:router(config-l2vpn)#pw-class span
RP/0/RP0/CPU0:router(config-l2vpn-pwc)#encapsulation mpls
RP/0/RP0/CPU0:router(config-l2vpn-pwc-mpls)#transport-mode ethernet
RP/0/RP0/CPU0:router(config-l2vpn)#xconnect group 1
RP/0/RP0/CPU0:router(config-l2vpn-xc)#p2p 2
RP/0/RP0/CPU0:router(config-l2vpn-xc-p2p)#monitor-session M1
RP/0/RP0/CPU0:router(config-l2vpn-xc-p2p)#neighbor ipv4 10.10.10.1 pw-id 2
RP/0/RP0/CPU0:router(config-l2vpn-xc-p2p)#pw-class span
RP/0/RP0/CPU0:router(config-l2vpn-xc-p2p)#commit
```

Verify SPAN over Pseudowire

The following examples show how to verify SPAN over Pseudowire configuration.

To check monitor session status:

```
RP/0/RP0/CPU0:router#show run monitor-session M1
monitor-session M1 ethernet
 destination pseudowire

RP/0/RP0/CPU0:router#show monitor-session M1 status
Monitor-session M1
Destination pseudowire
Source Interface      Dir   Status
BE100 (port)          Both  Operational
BE400 (port)          Both  Operational

RP/0/RP0/CPU0:router#show monitor-session M1 status detail
Monitor-session M1
  Destination pseudowire
  Source Interfaces
  -----------------
  Bundle-Ether100
    Direction:  Both
    Port level: True
    ACL match:  Disabled
    Portion:    Full packet
    Interval:   Mirror all packets
    Status:     Operational
  Bundle-Ether400
    Direction:  Both
    Port level: True
    ACL match:  Disabled
    Portion:    Full packet
    Interval:   Mirror all packets
    Status:     Operational
```

To check underlying l2vpn xconnect:

```
RP/0/RP0/CPU0:router#show run l2vpn
l2vpn
 pw-class span
  encapsulation mpls
   transport-mode ethernet
  !
  !
  p2p 2
   monitor-session M1
   neighbor ipv4 10.10.10.1 pw-id 2
   pw-class span
   !
  !
  p2p 10
```

```
   monitor-session M2
   neighbor ipv4 10.10.10.1 pw-id 10
    pw-class span
   !
  !
 !
!
RP/0/RP0/CPU0:router#show l2vpn xconnect
Fri Sep  6 03:41:15.691 UTC
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
        SB = Standby, SR = Standby Ready, (PP) = Partially Programmed

XConnect                   Segment 1                       Segment 2
Group      Name       ST   Description            ST       Description            ST
------------------------   ----------------------------    -----------------------------
1          2          UP   M1                     UP       10.10.10.1        2       UP

----------------------------------------------------------------------------------------
1          10         UP   M2                     UP       10.10.10.1        10      UP

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

Traffic Mirroring for Incoming and Outgoing Traffic Separately over Pseudowire

Table 5: Feature History Table

Feature Name	Release	Description
Traffic Mirroring for Incoming and Outgoing Traffic Separately over Pseudowire	Release 7.11.1	You can now distribute the monitoring load by separating the Rx and Tx traffic mirroring over the pseudowire. Earlier, you could mirror the entire traffic without distinguishing between Rx and Tx directions. The separation of traffic direction gives the flexibility of monitoring and analyzing the nature of data being sent and received using independent network traffic analysis tools. The separation also helps in distributing the monitoring load and eases troubleshooting. The feature modifies the **monitor-session** command. The keywords **destination rx** and **destination tx** of the command are extended to monitor session configuration mode. Earlier, this configuration resulted in verification failure.

Pseudowire Traffic Mirroring also known as PW-SPAN involves replicating designated traffic from the source port to a central location through the pseudowire. The transmission within the pseudowire follows a unidirectional flow, originating from the source port and terminating at the destination network analyzer.

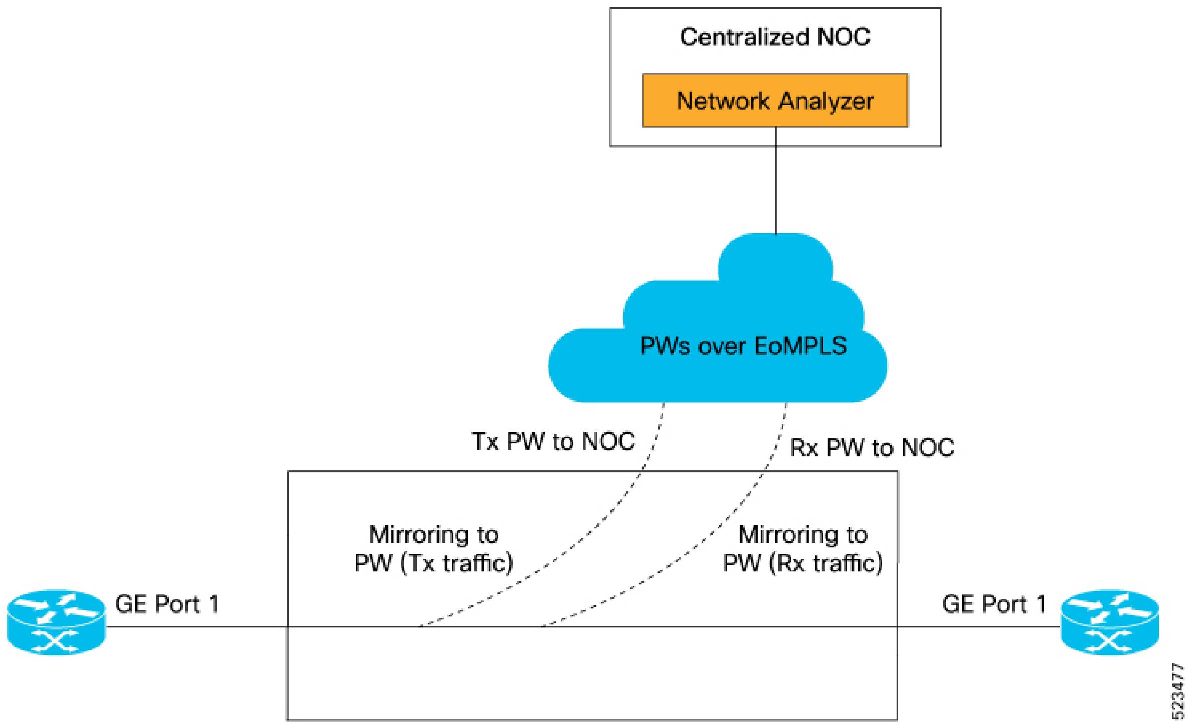
Previously, you could not send Rx and Tx mirrored traffic to separate Rx and Tx PW-SPAN destinations. The entire traffic is mirrored to the destination through pseudowire, which is less effective in monitoring and troubleshooting network issues. Resource allocation of monitoring tools is also not optimized, especially when the monitoring requirement for one direction is different from the other direction.

This feature allows separate Rx and Tx mirror destinations within a single session to optimize resource allocation when the monitoring requirement for one direction is different from the other direction.

Topology

Using this topology, let's understand how incoming and outgoing traffic are mirrored separately over pseudowire.

Figure 3: Mirroring Topology



- This topology uses pseudowires provisioned over EoMPLS.
- Two pseudowires, Rx PW and Tx PW, mirror incoming (Rx) and outgoing (Tx) traffic separately to a centralized Network Operations Center (NOC).
- The network analyzer hosted in the NOC receives the separately mirrored traffic for analysis.

You can provision pseudowires using L2VPN point-to-point cross-connect. The SPAN session supports configuring the session ID and traffic direction to allow multiple mirror destinations within the same SPAN session. After you configure traffic mirroring, traffic is duplicated from the selected pseudowires to the specified destination port without affecting the normal traffic forwarding in the network.

The destination port or monitoring tool captures mirrored traffic from the specified pseudowires, facilitating pseudowire traffic monitoring, analysis, and troubleshooting. The segregation of Rx and Tx monitoring enhances the ability to identify and isolate differences or performance problems. By identifying the root cause network problems can be resolved with greater efficiency and effectiveness.

Configure Traffic Mirroring for Incoming and Outgoing Traffic Separately over Pseudowire

Perform the following tasks to configure Rx and Tx pseudowire destinations:

- Create a pseudowire monitor session to replicate Ethernet traffic.
- Configure the destination for Rx and Tx traffic.
- Create an L2VPN cross-connect corresponding to the monitor session and define point-to-point forwarding details for Rx and Tx.
- Define bundle-ether interfaces for Rx and Tx directions.

```
Router(config)#monitor-session pw-span2 ethernet
Router(config-mon)#destination rx pseudowire
Router(config-mon)#destination tx pseudowire
Router(config-mon)#exit

Router(config)#l2vpn
Router(config-l2vpn)#xconnect group pw-span2
Router(config-l2vpn-xc)#p2p rx2
Router(config-l2vpn-xc-p2p)#monitor-session pw-span2 rx
Router(config-l2vpn-xc-p2p)#neighbor ipv4 100.2.1.11 pw-id 21
Router(config-l2vpn-xc-p2p-pw)#mpls static label local 1421 remote 1521
Router(config-l2vpn-xc-p2p-pw)#pw-class pw
Router(config-l2vpn-xc-p2p-pw)#exit
Router(config-l2vpn-xc-p2p)#exit
Router(config-l2vpn-xc)#p2p tx2
Router(config-l2vpn-xc-p2p)#monitor-session pw-span2 tx
Router(config-l2vpn-xc-p2p)#neighbor ipv4 100.1.1.22 pw-id 22
Router(config-l2vpn-xc-p2p-pw)#mpls static label local 1422 remote 1522
Router(config-l2vpn-xc-p2p-pw)#pw-class pw
Router(config-l2vpn-xc-p2p-pw)#exit
Router(config-l2vpn-xc-p2p)#exit
Router(config-l2vpn-xc)#exit
Router(config-l2vpn)#exit

Router(config)#interface Bundle-Ether1
Router(config-if)#ipv4 address 20.1.1.1 255.255.255.252
Router(config-if)#ipv6 address abc::20:1:1:1/126
Router(config-if)#lacp mode active
Router(config-if)#lacp period short

Router(config-if)#monitor-session pw-span2
Router(config-if-mon)#exit
Router(config-if)#exit

Router(config)#interface Bundle-Ether101
Router(config-if)#ipv4 address 20.1.4.1 255.255.255.252
Router(config-if)#ipv6 address abc::20:1:4:1/126
Router(config-if)#lacp mode active
Router(config-if)#lacp period short

Router(config-if)#monitor-session pw-span2
Router(config-if-mon)#exit
Router(config-if)#exit
Router(config-if)#load-interval 30
Router(config)#exit
```

Running Configuration

The following example shows the running configuration.

```
Router#sh run monitor-session pw-span2
Wed Sep 23 11:06:28.607 UTC
monitor-session pw-span2 ethernet
 destination rx pseudowire
 destination tx pseudowire
!

Router#sh run l2vpn xconnect group pw-span2
!l2vpn
l2vpn
 xconnect group pw-span2
  p2p rx2
   monitor-session pw-span2 rx
   neighbor ipv4 100.2.1.11 pw-id 21
    mpls static label local 1421 remote 1521
    pw-class pw
   !
  !
  p2p tx2
   monitor-session pw-span2 tx
   neighbor ipv4 100.1.1.22 pw-id 22
    mpls static label local 1422 remote 1522
    pw-class pw
   !
  !
 !
!

Router#sh run interface bundle-ether 1
interface Bundle-Ether1
 ipv4 address 20.1.1.1 255.255.255.252
 ipv6 address abc::20:1:1:1/126
 lacp mode active
 lacp period short
 monitor-session pw-span2
 !
!

Router#sh run interface bundle-ether 101
interface Bundle-Ether101
 ipv4 address 20.1.4.1 255.255.255.252
 ipv6 address abc::20:1:4:1/126
 lacp mode active
 lacp period short
 monitor-session pw-span2
 !
 load-interval 30
!
```

Verification

Verify that both Rx and Tx traffic is operational.

```
show monitor-session status
Monitor-session pw-span2
rx destination pseudowire
tx destination pseudowire
================================================================================
Source Interface      Dir   Status
--------------------- ----  ----------------------------------------------------
BE1                   both    Operational
BE101                 both    Operational
```

SPAN-to-File

SPAN-to-File is an extension of the pre-existing SPAN feature that allows network packets to be mirrored to a file instead of an interface. This simplifies the analysis of the packets at a later stage. The file format is PCAP, which helps that data to be used by tools, such as tcpdump or Wireshark.

Warning Be cautious when you apply this feature to files located on interfaces with high traffic.

When a file is configured as a destination for a SPAN session, a buffer is created on each node to which the network packets are logged. The buffer is for all packets on the node regardless of which interface they are from, that is, multiple interfaces may be providing packets for the same buffer. The buffers are deleted when the session configuration is removed. The file is written by each node to a location on the active RP which contains the node ID of the node on which the buffer was located.

If multiple interfaces are attached to a session, then interfaces on the same node are expected to have their packets sent to the same file. Bundle interfaces can be attached to a session with a file destination, which is similar to attaching individual interfaces.

SPAN-to-File Enhancements

Table 6: Feature History Table

Configure SPAN-to-File

Use the following command to configure SPAN to File:

```
monitor-session <name> [ethernet|ipv4|ipv6|mpls-ipv4|mpls-ipv6]
    destination file [size <kbytes>] [buffer-type linear]
```

The `monitor-session <name> [ethernet|ipv4|ipv6|mpls-ipv4|mpls-ipv6]` part of the command creates a monitor-session with the specified name and class and is a pre-existing chain point from the current SPAN feature. The `destination file [size <kbytes>] [buffer-type linear]` part of the command adds a new “file” option to the existing “destination”.

`destination file` has the following configuration options:

- Buffer size.
- Two types of buffer:
  - Circular: Once the buffer is full, the start is overwritten.
  - Linear: Once the buffer is full, no further packets are logged.

Note The default buffer-type is circular. Only linear buffer is explicitly configurable. Changing any of the parameters (buffer size or type) recreates the session, and clears any buffers of packets.

All configuration options which are applied to an attachment currently supported for other SPAN types should also be supported by SPAN to file. This may include:

- ACLs

- Write only first X bytes of packet.
- Mirror interval from 512 to 16k.

Note These options are implemented by the platform when punting the packet.

Once a session has been created, then interfaces may be attached to it using the following configuration:

```
interface GigabitEthernet 0/0/0/0
    monitor-session <name> [ethernet|ipv4|ipv6|mpls-ipv4|mpls-ipv6]
```

The attachment configuration is unchanged by SPAN-to-File feature.

Note Once the SPAN-to-File session is attached to source interface, mirroring starts and packets are punted from NPU to CPU and dropped at CPU until the **packet-collection start action** command is executed.

Configuration Examples

To configure a `mon1` monitor session, use these commands:

```
monitor-session mon1 ethernet
          destination file size 230000
         !
```

In the above example, omitting the `buffer-type` option results in default circular buffer.

To configure a `mon2` monitor session with the `linear` buffer type, use these commands:

```
monitor-session mon2 ethernet
          destination file size 1000 buffer-type linear
         !
```

To attach monitor session to a physical or bundle interface, use these commands:

```
interface Bundle-Ether1
monitor-session ms7 ethernet
!
```

Running Configuration

```
!! IOS XR Configuration 7.1.1.124I
!! Last configuration change at Tue Nov 26 19:29:05 2019 by root
!
hostname OC
logging console informational
!
monitor-session mon2 ethernet
 destination file size 1000 buffer-type linear

!
interface Bundle-Ether1
monitor-session ms7 ethernet
end
```

Verification

To verify packet collection status:

```
RP/0/RP0/CPU0:router#show monitor-session status
Monitor-session mon1
```

```
Destination File - Packet collecting
=============================================
Source Interface      Dir   Status
--------------------- ----  ------------------
Hu0/9/0/2             Rx    Operational

Monitor-session mon2
Destination File - Packet collecting
=========================================
Source Interface      Dir   Status
--------------------- ----  --------------
BE2.1                 Rx    Operational
```

If packet collection is not active, the following line is displayed:

```
Monitor-session mon2
Destination File - Not collecting
```

Here, `Status-Operational` and `Destination File - Not collecting` indicates that mirroring has started and packets are being punted from NPU to CPU but getting dropped at CPU until the **packet-collection start action** command is executed.

Action Commands for SPAN-to-File

Action commands are added to start and stop network packet collection. The commands may only be run on sessions where the destination is a file. The action command auto-completes names of globally configured SPAN to File sessions. See the table below for more information on action commands.

Table 7: Action Commands for SPAN-to-File

Action	Command	Description
Start	`monitor-session <name> packet-collection start`	Issue this command to start writing packets for the specified session to the configured buffer. Once the SPAN is configured and operational, the packets are punted to CPU and dropped by CPU until the `monitor-session <name> packet-collection start` command is executed.

Action	Command	Description
Stop	`monitor-session <name> packet-collection stop [ discard-data | write directory <dir> filename <filename> ]`	Issue this command to stop writing packets to the configured buffer. • `discard-data`: Specify this option to clear the buffer. • `discard-data`: Specify this option to write the buffer to the disk before it is cleared. The buffer is written in .pcap format in this location: `/<directory>/<node_id>/<filename>.pcap`. The .pcap extension that the user adds to the filename is removed automatically to avoid a duplicate file extension.

File Mirroring

Prior to Cisco IOS XR Software Release , the router did not support file mirroring from active RP to standby RP. Administrators had to manually perform the task or use EEM scripts to sync files across active RP and standby RP. Starting with Cisco IOS XR Software Release , the file mirroring feature enables the router to copy files or directories automatically from `/harddisk:/mirror` location in active RP to `/harddisk:/mirror` location in standby RP or RSP without user intervention or EEM scripts.

Two new CLIs have been introduced for the file mirroring feature:

- **mirror enable**

  The `/harddisk:/mirror` directory is created by default, but file mirroring functionality is only enabled by executing the `mirror enable` command from configuration terminal. Status of the mirrored files can be viewed with `show mirror status` command.

- **mirror enable checksum**

  The `mirror enable checksum` command enables MD5 checksum across active to standby RP to check integrity of the files. This command is optional.

Configure File Mirroring

File mirroring has to be enabled explicitly on the router. It is not enabled by default.

```
RP/0/RSP0/CPU0:router#show run mirror

Thu Jun 25 10:12:17.303 UTC
mirror enable
mirror checksum
```

Following is an example of copying running configuration to `harddisk:/mirror` location:

```
RP/0/RSP0/CPU0:router#copy running-config harddisk:/mirror/run_config
Wed Jul  8 10:25:51.064 PDT
Destination file name (control-c to abort): [/mirror/run_config]?
Building configuration..
```

```
32691 lines built in 2 seconds (16345)lines/sec
[OK]
```

Verification

To verify the syncing of file copied to mirror directory, use the `show mirror` command.

```
RP/0/RSP0/CPU0:router#show mirror
Wed Jul  8 10:31:21.644 PDT
% Mirror rsync is using checksum, this show command may take several minutes if you have
many files. Use Ctrl+C to abort
MIRROR DIR: /harddisk:/mirror/
% Last sync of this dir ended at Wed Jul  8 10:31:11 2020
Location   |Mirrored |MD5 Checksum                     |Modification Time
---------------------------------------------------------------------------
run_config |yes      |76fc1b906bec4fe08ecda0c93f6c7815 |Wed Jul  8 10:25:56 2020
```

If checksum is disabled, `show mirror` command displays the following output:

```
RP/0/RSP0/CPU0:router#show mirror
Wed Jul 8 10:39:09.646 PDT
MIRROR DIR: /harddisk:/mirror/
% Last sync of this dir ended at Wed Jul  8 10:31:11 2020
Location   |Mirrored |Modification Time
-----------------------------------------
run_config |yes         |Wed Jul  8 10:25:56 2020
```

If there is a mismatch during the syncing process, use `show mirror mismatch` command to verify.

```
RP/0/RP0/CPU0:router# show mirror mismatch
Wed Jul  8 10:31:21.644 PDT
 MIRROR DIR: /harddisk:/mirror/
 % Last sync of this dir ended at Wed Jul  8 10:31:11 2020
 Location  |Mismatch Reason     |Action Needed
------------------------------------------------
 test.txt  |newly created item. |send to standby
```

Troubleshoot Traffic Mirroring

When you encounter any issue with traffic mirroring, begin troubleshooting by checking the output of the **show monitor-session status** command. This command displays the recorded state of all sessions and source interfaces:

```
# show monitor-session status
Monitor-session 5
rx destination interface tunnel-ip5
tx destination is not specified
=================================================================================
Source Interface  Dir  Status
--------------------- ---- ----------------------------------------------------
Te0/0/0/23 (port) Rx   Operational
```

In the preceding example, the line marked as `<Session status>` can indicate one of these configuration errors:

Session Status	Explanation
Session is not configured globally	The session does not exist in global configuration. Review the command output and ensure that a session with a correct name configured.

Session Status	Explanation
Destination interface <intf> (<down-state>)	The destination interface is not in Up state in the Interface Manage can verify the state using the **show interfaces** command. Check th configuration to determine what might be keeping the interface from up (for example, a sub-interface needs to have an appropriate encaps configured).

The <Source interface status> can report these messages:

Source Interface Status	Explanation
Operational	Everything appears to be working correctly in traffic mirroring PI. follow up with the platform teams in the first instance, if mirroring operating as expected.
Not operational (Session is not configured globally)	The session does not exist in global configuration. Check the **show** command output to ensure that a session with the right name has b configured.
Not operational (destination not known)	The session exists, but it either does not have a destination interface s or the destination interface named for the session does not exist. For e if the destination is a sub-interface that has not been created.
Not operational (source same as destination)	The session exists, but the destination and source are the same inte traffic mirroring does not work.
Not operational (destination not active)	The destination interface or pseudowire is not in the Up state. See corresponding *Session status* error messages for suggested resoluti
Not operational (source state <down-state>)	The source interface is not in the Up state. You can verify the state the **show interfaces** command. Check the configuration to see wha be keeping the interface from coming up (for example, a sub-interfa to have an appropriate encapsulation configured).
Error: see detailed output for explanation	Traffic mirroring has encountered an error. Run the **show monitor status detail** command to display more information.

The **show monitor-session status detail** command displays full details of the configuration parameters and any errors encountered. For example:

RP/0/RP0/CPU0:router **show monitor-session status detail**

```
Monitor-session sess1
 Destination interface is not configured
 Source Interfaces
 -----------------
 TenGigE0/0/0/1
  Direction: Both
  ACL match: Disabled
  Portion:  Full packet
  Status:  Not operational (destination interface not known)
 TenGigE0/0/0/2
  Direction: Both
  ACL match: Disabled
  Portion:  First 100 bytes
```

```
  Status: Not operational (destination interface not known). Error: 'Viking SPAN PD' detected
 the 'warning' condition 'PRM connection
          creation failure'.
Monitor-session foo
 Destination next-hop TenGigE 0/0/0/0
 Source Interfaces
 -----------------
 TenGigE 0/0/0/1.100:
  Direction: Both
  Status:  Operating
 TenGigE 0/0/0/2.200:
  Direction: Tx
  Status:  Error: <blah>

Monitor session bar
 No destination configured
 Source Interfaces
 -----------------
 TenGigE 0/0/0/3.100:
  Direction: Rx
  Status:  Not operational(no destination)
```

Here are additional trace and debug commands:

```
RP/0/RP0/CPU0:router# show monitor-session trace ?

 platform  Enable platform trace
 process   Filter debug by process(cisco-support)

RP/0/RP0/CPU0:router# show monitor-session trace platform ?

 errors  Display error traces(cisco-support)
 events  Display event traces(cisco-support)

RP/0/RP0/CPU0:router#show monitor-session trace platform events location all ?

usrtdir  Specify directory to collect unsorted traces(cisco-support)
|        Output Modifiers
<cr>

RP/0/RP0/CPU0:router#show monitor-session trace platform errors location all ?

usrtdir  Specify directory to collect unsorted traces(cisco-support)
|        Output Modifiers
<cr>

#



RP/0/RP0/CPU0:router# debug monitor-session process all

RP/0/RP0/CPU0:router# debug monitor-session process ea

RP/0/RP0/CPU0:router# debug monitor-session process ma

RP/0/RP0/CPU0:router# show monitor-session process mgr

 detail  Display detailed output
 errors  Display only attachments which have errors
 internal Display internal monitor-session information
 |     Output Modifiers
```

```
RP/0/RP0/CPU0:router# show monitor-session status

RP/0/RP0/CPU0:router# show monitor-session status errors

RP/0/RP0/CPU0:router# show monitor-session status internal

RP/0/RP0/CPU0:router# show tech-support span ?

 file       Specify a valid file name (e.g. disk0:tmp.log)
 list-CLIs  list the commands that would be run (don't execute)(cisco-support)
 location   Specify a location(cisco-support)
 rack       Specify a rack(cisco-support)
 time-out   per show command timeout configuration(cisco-support)
 <cr>
```