Juniper Networks | Engineering Simplicity

# Junos® OS

# Tunnel and Encryption Services Interfaces User Guide for Routing Devices

Published
2023-06-13

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



## YEAR 2000 NOTICE

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

## END USER LICENSE AGREEMENT

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

# Table of Contents

## 4 Configuration Statements: Generic Routing Encapsulation (GRE) Tunnel, IPIP, and UDP Encapsulation and Decapsulation using Flexible Tunnel Interfaces (FTIs)



## 5 Operational Commands

# About This Guide

Use this guide to configure and monitor tunneling, which encapsulates packets inside a transport protocol, providing a private, secure path through an otherwise public network.

1

CHAPTER

# Tunnel Services

# Tunnel Services Overview

IN THIS SECTION



## Tunnel Services Overview

By encapsulating arbitrary packets inside a transport protocol, tunneling provides a private, secure path through an otherwise public network. Tunnels connect discontinuous subnetworks and enable encryption interfaces, virtual private networks (VPNs), and MPLS. If you have a Tunnel *Physical Interface Card* (PIC) installed in your M Series or T Series router, you can configure unicast, multicast, and logical tunnels.

You can configure two types of tunnels for VPNs: one to facilitate routing table lookups and another to facilitate VPN routing and forwarding instance (VRF) table lookups.

For information about encryption interfaces, see "Configuring Encryption Interfaces" on page 144. For information about VPNs, see the Junos OS VPNs Library for Routing Devices. For information about MPLS, see the MPLS Applications User Guide.

On SRX Series Firewalls, Generic Routing Encapsulation (GRE) and IP-IP tunnels use internal interfaces, gr-0/0/0 and ip-0/0/0, respectively. The Junos OS creates these interfaces at system bootup; they are not associated with physical interfaces.

The Juniper Networks Junos OS supports the tunnel types shown in the following table.

**Table 1: Tunnel Interface Types**

| Interface | Description |
|---|---|
| `gr-0/0/0` | Configurable generic routing encapsulation (GRE) interface. GRE allows the encapsulation of one routing protocol over another routing protocol.<br><br>Within a router, packets are routed to this internal interface, where they are first encapsulated with a GRE packet and then re-encapsulated with another protocol packet to complete the GRE. The GRE interface is an internal interface only and is not associated with a physical interface. You must configure the interface for it to perform GRE. |
| `gre` | Internally generated GRE interface. This interface is generated by the Junos OS to handle GRE. You cannot configure this interface. |
| `ip-0/0/0` | Configurable IP-over-IP encapsulation (also called IP tunneling) interface. IP tunneling allows the encapsulation of one IP packet over another IP packet.<br><br>Packets are routed to an internal interface where they are encapsulated with an IP packet and then forwarded to the encapsulating packet's destination address. The IP-IP interface is an internal interface only and is not associated with a physical interface. You must configure the interface for it to perform IP tunneling. |
| `ipip` | Internally generated IP-over-IP interface. This interface is generated by the Junos OS to handle IP-over-IP encapsulation. It is not a configurable interface. |
| `lt-0/0/0` | The `lt` interface on M Series and T Series routers supports configuration of logical systems—the capability to partition a single physical router into multiple logical devices that perform independent routing tasks.<br><br>On SRX Series Firewalls, the `lt` interface is a configurable logical tunnel interface that interconnects logical systems. See the *Junos OS Logical Systems Configuration Guide for Security Devices*. |

**Table 1: Tunnel Interface Types** ***(Continued)***

| Interface | Description |
|---|---|
| mt-0/0/0 | Internally generated multicast tunnel interface. Multicast tunnels filter all unicast packets; if an incoming packet is not destined for a 224/8-or-greater prefix, the packet is dropped and a counter is incremented.<br><br>Within a router, packets are routed to this internal interface for multicast filtering. The multicast tunnel interface is an internal interface only and is not associated with a physical interface. If your router has a Tunnel Services PIC, the Junos OS automatically configures one multicast tunnel interface (mt-) for each virtual private network (VPN) you configure. You do not need to configure multicast tunnel interfaces. However, you can configure properties on mt- interfaces, such as the multicast-only statement. |
| mtun | Internally generated multicast tunnel interface. This interface is generated by the Junos OS to handle multicast tunnel services. It is not a configurable interface. |
| pd-0/0/0 | Configurable Protocol Independent Multicast (PIM) de-encapsulation interface. In PIM sparse mode, the first-hop router encapsulates packets destined for the rendezvous point router. The packets are encapsulated with a unicast header and are forwarded through a unicast tunnel to the rendezvous point. The rendezvous point then de-encapsulates the packets and transmits them through its multicast tree.<br><br>Within a router, packets are routed to this internal interface for de-encapsulation. The PIM de-encapsulation interface is an internal interface only and is not associated with a physical interface. You must configure the interface for it to perform PIM de-encapsulation.<br><br>**NOTE**: On SRX Series Firewalls, this interface type is ppd0. |
| pe-0/0/0 | Configurable PIM encapsulation interface. In PIM sparse mode, the first-hop router encapsulates packets destined for the rendezvous point router. The packets are encapsulated with a unicast header and are forwarded through a unicast tunnel to the rendezvous point. The rendezvous point then de-encapsulates the packets and transmits them through its multicast tree.<br><br>Within a router, packets are routed to this internal interface for encapsulation. The PIM encapsulation interface is an internal interface only and is not associated with a physical interface. You must configure the interface for it to perform PIM encapsulation.<br><br>**NOTE**: On SRX Series Firewalls, this interface type is ppe0. |

**Table 1: Tunnel Interface Types *(Continued)***

| Interface | Description |
|---|---|
| pimd | Internally generated PIM de-encapsulation interface. This interface is generated by the Junos OS to handle PIM de-encapsulation. It is not a configurable interface. |
| pime | Internally generated PIM encapsulation interface. This interface is generated by the Junos OS to handle PIM encapsulation. It is not a configurable interface. |
| vt-0/0/0 | Configurable virtual loopback tunnel interface. Facilitates VRF table lookup based on MPLS labels. This interface type is supported on M Series and T Series routers, but not on SRX Series Firewalls.<br><br>To configure a virtual loopback tunnel to facilitate VRF table lookup based on MPLS labels, you specify a virtual loopback tunnel interface name and associate it with a routing instance that belongs to a particular routing table. The packet loops back through the virtual loopback tunnel for route lookup. |

Starting in Junos OS Release 15.1, you can configure Layer 2 Ethernet services over GRE interfaces (gr-*fpc/pic/port* to use GRE encapsulation). To enable Layer 2 Ethernet packets to be terminated on GRE tunnels, you must configure the bridge domain protocol family on the gr- interfaces and associate the gr- interfaces with the bridge domain. You must configure the GRE interfaces as core-facing interfaces, and they must be access or trunk interfaces. To configure the bridge domain family on gr- interfaces, include the family bridge statement at the [edit interfaces gr-*fpc/pic/port* unit *logical-unit-number*] hierarchy level. To associate the gr- interface with a bridge domain, include the interface gr-*fpc/pic/port* statement at the [edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name*] hierarchy level. You can associate GRE interfaces in a bridge domain with the corresponding VLAN ID or list of VLAN IDs in a bridge domain by including the vlan-id (all | none | number) statement or the vlan-id-list [ *vlan-id-numbers* ] statement at the [edit bridge-domains *bridge-domain-name*] hierarchy level. The VLAN IDs configured for the bridge domain must match with the VLAN IDs that you configure for GRE interfaces by using the vlan-id (all | none | number) statement or the vlan-id-list [ *vlan-id-numbers* ] statement at the [edit interfaces gr-*fpc/pic/port* unit logical-unit-number] hierarchy level. You can also configure GRE interfaces within a bridge domain associated with a virtual switch instance. Layer 2 Ethernet packets over GRE tunnels are also supported with the GRE key option. The gre-key match condition allows a user to match against the GRE key field, which is an optional field in GRE encapsulated packets. The key can be matched as a single key value, a range of key values, or both.

**NOTE**: Starting in Junos OS Release 16.1, Layer 2 Port mirroring to a remote collector over a GRE interface is supported.

SEE ALSO



# Tunnel Interfaces on MX Series Routers with Line Cards (MPC7E through MPC11E)

IN THIS SECTION



MPC7E-10G, MPC7E-MRATE, MX2K-MPC8E, and MX2K-MPC9E support a total of four inline tunnel interfaces per MPC, one per PIC. You can create a set of tunnel interfaces per PIC slot up to a maximum of four slots (from 0 through 3) on MX Series routers with these MPCs.

MPC10E-15C supports three inline tunnel interfaces per MPC, one per PIC, whereas MPC10E-10C supports two inline tunnel interfaces per MPC, one per PIC. On MX Series routers with MPC10E-15C, you can

create a set of tunnel interfaces per PIC slot up to a maximum of three slots (from 0 through 2). And, on MX Series routers with MPC10E-10C, you can create a set of tunnel interfaces per PIC slot up to a maximum of two slots (0 and 1).

MX2K-MPC11E supports 8 inline tunnel interfaces per MPC, one per PIC. On MX Series routers with MX2K-MPC11E, you can create a set of tunnel interfaces per PIC slot up to a maximum of eight slots (from 0 through 7). These PICs are referred to as pseudo tunnel PICs. You create tunnel interfaces on MX Series routers with MPC7E-10G, MPC7E-MRATE, MX2K-MPC8E, MX2K-MPC9E, MPC10E-15C, MPC10E-10C, and MX2K-MPC11E by including the following statements at the **[edit chassis]** hierarchy level:

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth ;
        }
    }
}
```

## Packet Forwarding Engine Mapping and Tunnel Bandwidth for MPC7E-MRATE

The tunnel bandwidth for MPC7E-MRATE is 1–120Gbps with an increment of 1Gbps. However, if you do not specify the bandwidth in the configuration, it is set to 120Gbps.

Table 2 on page 7 shows the mapping between the tunnel bandwidth and the Packet Forwarding Engines for MPC7-MRATE .

**Table 2: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MPC7E-MRATE**

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | PFE Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC0 | 120Gbps | PFE0 | 120Gbps | 240Gbps |
| PIC1 | 120Gbps | | | |
| PIC2 | 120Gbps | PFE1 | 120Gbps | 240Gbps |
| PIC3 | 120Gbps | | | |

## Packet Forwarding Engine Mapping and Tunnel Bandwidth for MPC7E-10G

The tunnel bandwidth for MPC7E-10G is 1–120Gbps with an incrementof 1Gbps However, if you do not specify the bandwidth in the configuration, it is set to 120Gbps.

Table 3 on page 8 shows the mapping between the tunnel bandwidth and the Packet Forwarding Engines for MPC7E-10G.

**Table 3: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MPC7E-10G**

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | PFE Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC0 | 120Gbps | PFE0 | 120Gbps | 200Gbps |
| PIC1 | 120Gbps | | | |
| PIC2 | 120Gbps | PFE1 | 120Gbps | 200Gbps |
| PIC3 | 120Gbps | | | |

## Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX2K-MPC8E

The tunnel bandwidth for MX2K-MPC8E is 1– 120Gbps with an increment of 1Gbps. However, if you do not specify the bandwidth in the configuration, it is set to 120Gbps.

Table 4 on page 8 shows the mapping between the tunnel bandwidth and the Packet Forwarding Engines for MX2K-MPC8E.

**Table 4: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX2K-MPC8E**

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | Packet Forwarding Engine Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC0 | 120Gbps | PFE0 | 120Gbps | 240Gbps |
| PIC1 | 120Gbps | PFE1 | 120Gbps | 240Gbps |
| PIC2 | 120Gbps | PFE2 | 120Gbps | 240Gbps |
| PIC3 | 120Gbps | PFE3 | 120Gbps | 240Gbps |

## Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX2K-MPC9E

The tunnel bandwidth for MX2K-MPC9E is 1–200Gbps with an increment of 1Gbps However, if you do not specify the bandwidth in the configuration, it is set to 200Gbps.

Table 5 on page 9 shows the mapping between the tunnel bandwidth and the Packet Forwarding Engines for MX2K-MPC9E.

**Table 5: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX2K-MPC9E**

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | PFE Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC0 | 200Gbps | PFE0 | 200Gbps | 400Gbps |
| PIC1 | 200Gbps | PFE1 | 200Gbps | 400Gbps |
| PIC2 | 200Gbps | PFE2 | 200Gbps | 400Gbps |
| PIC3 | 200Gbps | PFE3 | 200Gbps | 400Gbps |

## Packet Forwarding Engine Mapping and Tunnel Bandwidth for MPC10E-10C

The tunnel bandwidth for MPC10E-10C is 1–400Gbps with an increment of 1Gbps. However, if you do not specify the bandwidth in the configuration, it is set to 400Gbps.

Table 6 on page 9 shows the mapping between the tunnel bandwidth and the Packet Forwarding Engines for MPC10E-10C.

**Table 6: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MPC10E-10C.**

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | Packet Forwarding Engine Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC0 | 250Gbps | PFE0 | 250Gbps | 500Gbps |
| PIC1 | 250Gbps | PFE1 | 250Gbps | 500Gbps |

## Packet Forwarding Engine Mapping and Tunnel Bandwidth for MPC10E-15C

The tunnel bandwidth for MPC10E-15C is 1–400Gbps with an increment of 1Gbps. However, if you do not specify the bandwidth in the configuration, it is set to 400Gbps.

Table 7 on page 10 shows the mapping between the tunnel bandwidth and the Packet Forwarding Engines for MPC10E-15C.

**Table 7: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MPC10E-15C.**

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | Packet Forwarding Engine Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC0 | 250Gbps | PFE0 | 250Gbps | 500Gbps |
| PIC1 | 250Gbps | PFE1 | 250Gbps | 500Gbps |
| PIC2 | 250Gbps | PFE2 | 250Gbps | 500Gbps |

## Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX2K-MPC11E

The tunnel bandwidth for MX2K-MPC11E is 1–400Gbps with an increment of 1Gbps. However, if you do not specify the bandwidth in the configuration, it is set to 400Gbps.

Table 8 on page 10 shows the mapping between the tunnel bandwidth and the Packet Forwarding Engines for MX2K-MPC11E .

**Table 8: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX2K-MPC11E**

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | PFE Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC0 | 250Gbps | PFE0 | 250Gbps | 500Gbps |
| PIC1 | 250Gbps | PFE1 | 250Gbps | 500Gbps |
| PIC2 | 250Gbps | PFE2 | 250Gbps | 500Gbps |

**Table 8: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX2K-MPC11E *(Continued)***

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | PFE Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC3 | 250Gbps | PFE3 | 250Gbps | 500Gbps |
| PIC4 | 250Gbps | PFE4 | 250Gbps | 500Gbps |
| PIC5 | 250Gbps | PFE5 | 250Gbps | 500Gbps |
| PIC6 | 250Gbps | PFE6 | 250Gbps | 500Gbps |
| PIC7 | 250Gbps | PFE7 | 250Gbps | 500Gbps |

**NOTE**: An unspecified tunnel services bandwidth value in the configuration for MPC10E-10C, MPC10E-15C, and MX2K-MPC11E results in a value larger than the maximum tunnel bandwidth per PFE in certain traffic conditions.

## Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX10K-LC9600

The tunnel bandwidth for MX10K-LC9600 is 1–400Gbps with an increment of 1Gbps. However, if you do not specify the bandwidth in the configuration, it is set to 400Gbps.

Table 9 on page 11 shows the mapping between the tunnel bandwidth and the Packet Forwarding Engines for MX10K-LC9600.

**Table 9: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX10K-LC9600**

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | PFE Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC0 | 200Gbps | PFE0 | 200Gbps | 400Gbps |
| | | PFE1 | | |

**Table 9: Packet Forwarding Engine Mapping and Tunnel Bandwidth for MX10K-LC9600 *(Continued)***

| Pseudo Tunnel PIC | Maximum Bandwidth per Tunnel PIC | PFE Mapping | Maximum Tunnel Bandwidth per PFE | Maximum PFE Bandwidth |
|---|---|---|---|---|
| PIC1 | 200Gbps | PFE2 | 200Gbps | 400Gbps |
| | | PFE3 | | |
| PIC2 | 200Gbps | PFE4 | 200Gbps | 400Gbps |
| | | PFE5 | | |
| PIC3 | 200Gbps | PFE6 | 200Gbps | 400Gbps |
| | | PFE7 | | |
| PIC4 | 200Gbps | PFE8 | 200Gbps | 400Gbps |
| | | PFE9 | | |
| PIC5 | 200Gbps | PFE10 | 200Gbps | 400Gbps |
| | | PFE11 | | |

## SEE ALSO

*tunnel-services*

*bandwidth*

## Dynamic Tunnels Overview

A VPN that travels through a non-MPLS network requires a GRE tunnel. This tunnel can be either a static tunnel or a dynamic tunnel. A static tunnel is configured manually between two PE routers. A dynamic tunnel is configured using BGP route resolution.

When a router receives a VPN route that resolves over a BGP next hop that does not have an MPLS path, a GRE tunnel can be created dynamically, allowing the VPN traffic to be forwarded to that route. Only GRE IPv4 tunnels are supported.

To configure a dynamic tunnel between two PE routers, include the `dynamic-tunnels` statement:

```
dynamic-tunnels tunnel-name {
    destination-networks prefix;
    source-address address;
}
```

You can configure this statement at the following hierarchy levels:

- `[edit routing-options]`
- `[edit routing-instances` *`routing-instance-name`* `routing-options]`
- `[edit logical-systems` *`logical-system-name`* `routing-options]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `routing-options]`

#### SEE ALSO

*dynamic-tunnels*

Junos OS Routing Protocols Library

Junos OS VPNs Library for Routing Devices

**Release History Table**

| Release | Description |
|---|---|
| 16.1 | Starting in Junos OS Release 16.1, Layer 2 Port mirroring to a remote collector over a GRE interface is supported. |
| 15.1 | Starting in Junos OS Release 15.1, you can configure Layer 2 Ethernet services over GRE interfaces (`gr-`*`fpc/pic/port`* to use GRE encapsulation). |

# Configuring Tunnel Interfaces

IN THIS SECTION



## Tunnel Interface Configuration on MX Series Routers Overview

Because MX Series routers do not support Tunnel Services PICs, you create tunnel interfaces on MX Series routers by including the following statements at the `[edit chassis]` hierarchy level:

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth (1g | 10g | 20g | 30g | 40g | 50g | 60g | 70g | 80g | 90g | 100g);
        }
```

```
    }
}
```

Where:

`fpc` *`slot-number`* is the slot number of the DPC, MPC, or MIC. On the MX80 router, possible values are 0 and 1. On other MX Series routers, if two SCBs are installed, the range is 0 through 11. If three SCBs are installed, the range is 0 through 5 and 7 through 11.

`pic` *`number`* is the slot number of the PIC. On MX80 routers, if the FPC is 0, the PIC number can only be 0. If the FPC is 1, the PIC range is 0 through 3. For all other MX Series routers, the range is 0 through 3.

`bandwidth` (`1g` | `10g` | `20g` | `30g` | `40g` | `50g` | `60g` | `70g` | `80g` | `90g` | `100g`) is the maximum amount of bandwidth, in gigabits, that is available for tunnel traffic on each Packet Forwarding Engine. For MPCs and MICs, this bandwidth is not reserved for tunnel traffic and can be shared by the network interfaces. For DPCs, this bandwidth is reserved and cannot be shared by the network interfaces.

> **NOTE**: When you use MPCs and MICs, tunnel interfaces are soft interfaces and allow as much traffic as the forwarding-path allows, so it is advantageous to set up tunnel services without artificially limiting traffic by use of the `bandwidth` option. However, you *must* specify `bandwidth` when configuring tunnel services for MX Series routers with DPCs or FPCs. The GRE key option is not supported on the tunnel interfaces for DPCs on MX960 routers.

If you specify a bandwidth that is not compatible, tunnel services are not activated. For example, you cannot specify a bandwidth of 1 Gbps for a Packet Forwarding Engine on a 10-Gigabit Ethernet 4-port DPC.

When you configure tunnel interfaces on the Packet Forwarding Engine of a 10-Gigabit Ethernet 4-port DPC, the Ethernet interfaces for that port are removed from service and are no longer visible in the command-line interface (CLI). The Packet Forwarding Engine of a 10-Gigabit Ethernet 4-port DPC supports either tunnel interfaces or Ethernet interfaces, but not both. Each port on the 10-Gigabit Ethernet 4-port DPC includes two LEDs, one for tunnel services and one for Ethernet services, to indicate which type of service is being used. On the Gigabit Ethernet 40-port DPC, you can configure both tunnel and Ethernet interfaces at the same time.

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer. The bandwidth that you specify determines the port number of the tunnel interfaces that are created. When you specify a bandwidth of `1g`, the port number is always 10. When you specify any other bandwidth, the port number is always 0.

**NOTE**: When the tunnel bandwidth is unspecified in the Routing Engine CLI, the maximum tunnel bandwidth for an MPC3E is 60G.

**NOTE**: You cannot configure ingress queueing and tunnel services on the same MPC because doing so causes PFE forwarding to stop. You can configure and use each feature separately.

SEE ALSO

*bandwidth (Tunnel Services)*

*tunnel-services (Chassis)*

## Configuring Tunnel Interfaces on an MX Series Router with a 16x10GE 3D MPC

MX960, MX480, and M240 routers support the 16-port 10-Gigabit Ethernet MPC (16x10GE 3D MPC) fixed configuration Field Replaceable Unit (FRU). Each Packet Forwarding Engine on a 16x10GE MPC can support a full-duplex 10Gbps tunnel without losing line-rate capacity. For example, a full-duplex 10Gbps tunnel can be hosted on a 10-Gigabit-Ethernet port, while two other 10-Gigabit-Ethernet ports on the same PFE can concurrently forward line-rate traffic.

To configure an MPC and its corresponding Packet Forwarding Engine to use tunneling services, include the `tunnel-services` statement at the `[edit chassis fpc` *`slot-number`* `pic` *`pic-number`*`]` hierarchy level. The Junos OS creates tunnel interfaces **gr-*fpc*/*pic*/*port*.0**, **vt-*fpc*/*pic*/*port*.0**, and so on. You also configure the amount of bandwidth reserved for tunnel services.

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth 10g;
        }
    }
}
```

**fpc** ***slot-number*** is the slot number of the MPC. If two SCBs are installed, the range is 0 through 11. If three SCBs are installed, the range is 0 through 5 and 7 through 11.

**pic** ***number*** is the number of the Packet Forwarding Engine on the MPC. The range is 0 through 3.

**bandwidth 10g** is the amount of bandwidth to reserve for tunnel traffic on each Packet Forwarding Engine.

In the following example, you create tunnel interfaces on Packet Forwarding Engine 0 of MPC 4 with 10 Gbps of bandwidth reserved for tunnel traffic. With this configuration, the tunnel interfaces created are **gr-4/0/0**, **pe-4/0/0**, **pd-4/0/0**, **vt-4/0/0**, and so on.

```
[edit chassis]
fpc 4 pic 0 {
    tunnel-services {
        bandwidth 10g;
    }
}
```

#### SEE ALSO

Configuring Junos OS to Run a Specific Network Services Mode in MX Series Routers

## Configuring Tunnel Interfaces on MX Series Routers with the MPC3E

Because the MX Series routers do not support Tunnel Services PICs, you create tunnel interfaces on MX Series routers by including the following statements at the `[edit chassis]` hierarchy level:

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth (1g | 10g | 20g | 40g);
        }
    }
}
```

`fpc` *`slot-number`* is the slot number of the DPC, MPC, or MIC. On the MX80 router, the range is 0 through 1. On other MX series routers, if two SCBs are installed, the range is 0 through 11. If three SCBs are installed, the range is 0 through 5 and 7 through 11.

The `pic` *`number`* On MX80 routers, if the FPC is 0, the PIC number can only be 0. If the FPC is 1, the PIC range is 0 through 3. For all other MX series routers, the range is 0 through 3.

`bandwidth` (`1g` | `10g` | `20g` | `40g`) is the amount of bandwidth to reserve for tunnel traffic on each Packet Forwarding Engine.

> **NOTE**: When you use MPCs and MICs, tunnel interfaces are soft interfaces and allow as much traffic as the forwarding-path allows, so it is advantageous to setup tunnel services without artificially limiting traffic by use of the `bandwidth` option. However, you *must* specify `bandwidth` when configuring tunnel services for MX Series routers with DPCs or FPCs.

`1g` indicates that 1 gigabit per second of bandwidth is reserved for tunnel traffic.

`10g` indicates that 10 gigabits per second of bandwidth is reserved for tunnel traffic.

`20g` indicates that 20 gigabits per second of bandwidth is reserved for tunnel traffic.

`40g` indicates that 40 gigabits per second of bandwidth is reserved for tunnel traffic.

If you specify a bandwidth that is not compatible, tunnel services are not activated. For example, you cannot specify a bandwidth of 1 Gbps for a Packet Forwarding Engine on a 10-Gigabit Ethernet 4-port DPC.

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer. The bandwidth that you specify determines the port number of the tunnel interfaces that are created. When you specify a bandwidth of `1g`, the port number is always 10. When you specify any other bandwidth, the port number is always 0.

#### SEE ALSO

*bandwidth (Tunnel Services)*

*tunnel-services (Chassis)*

# Example: Configuring Tunnel Interfaces on the MPC3E

IN THIS SECTION



## Requirements for Configuration of Tunnel Interfaces on the MPC3E

This example requires MX Series routers with the MPC3E.

## Ethernet Tunnel Configuration Overview

MX Series routers do not support Tunnel Services PICs. However, you can create one set of tunnel interfaces per pic slot up to a maximum of 4 slots from 0-3 on MX Series routers with the MPC3E.

To configure the tunnels, include the **tunnel-services** statement and an optional bandwidth of (**1g** | **10g** | **20g** | **30g** | **40g**) at the **[edit chassis]** hierarchy level.

**NOTE**: When no tunnel bandwidth is specified, the tunnel interface can have a maximum bandwidth of up to 60Gbps.

**NOTE**: A MIC need not be plugged in to the MPC3E to configure a tunnel interface.

## Configuring a 20-Gigabit Ethernet Tunnel

IN THIS SECTION

#### Procedure

#### Step-by-Step Procedure

In the following example, you create tunnel interfaces on PIC-slot 1 of MPC 0 with 20 gigabit per second of bandwidth reserved for tunnel traffic. With this configuration, the tunnel interfaces created are **gr-0/1/0**, **pe-0/1/0**, **pd-0/1/0**, **vt-0/1/0**, and so on.

1. To create a 20 gigabit per second tunnel interface, use the following configuration:

```
[edit chassis]
fpc 0 pic 1 {
    tunnel-services {
        bandwidth 20g;
    }
}
```

## Configuring a Tunnel With Unspecified Bandwidth

**IN THIS SECTION**



#### Procedure

#### Step-by-Step Procedure

In the following example, you create a tunnel interface on PIC-slot 3 of MPC 0 with no bandwidth specified. The tunnel traffic can carry up to a maximum of 60Gbps depending on other trafic through the packet forwarding engine. With this configuration, the tunnel interfaces created are **gr-0/3/0**, **pe-0/3/0**, **pd-0/3/0**, **vt-0/3/0**, and so on.

1. To create a tunnel interface with no bandwidth specification, use the following configuration:

```
[edit chassis]
fpc 0 pic 3 {
    tunnel-services;
}
```

SEE ALSO

*bandwidth (Tunnel Services)*

*tunnel-services (Chassis)*

## Configuring Tunnel Interfaces on MX Series Routers with MPC4E

MX Series routers do not support Tunnel Services PICs. However, you can create a set of tunnel interfaces per PIC slot up to a maximum of four slots from 0 through 3 on MX Series routers with MPC4E.

To configure the tunnel interfaces, include the `tunnel-services` statement and an optional bandwidth of (`1g | 10g | 20g | 30g | 40g`) at the `[edit chassis]` hierarchy level. When no tunnel bandwidth is specified, the tunnel interface can have a maximum bandwidth of up to 60 Gbps.

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer. The bandwidth that you specify determines the port number of the tunnel interfaces that are created. When you specify a bandwidth of `1g`, the port number is always 10. When you specify any other bandwidth, the port number is always 0.

In the following example, you create tunnel interfaces on **PIC 1** of **MPC 4** with 40 Gbps of bandwidth reserved for tunnel traffic. `fpc` *`slot-number`* is the slot number of the MPC. In this configuration, the tunnel interfaces created are gr-4/1/1, pe-4/1/1, pd-4/1/1, vt-4/1/1, and so on.

To create a 40-Gbps tunnel interface, use the following configuration:

```
[edit chassis]
fpc 4 pic 1 {
    tunnel-services {
        bandwidth 40g;
    }
}
```

SEE ALSO

*bandwidth (Tunnel Services)*

*tunnel-services (Chassis)*

## Configuring Tunnel Interfaces on MX Series Routers with MPC7E-MRATE/MPC7E-10G

MPCs support a total of four inline tunnels per MPC, one per PIC. You can create a set of tunnel interfaces per PIC slot up to a maximum of four slots from 0 through 3

To configure the tunnel interfaces, include the `tunnel-services` statement and an optional bandwidth of 1 Gbps through 120Gbps at the `[edit chassis fpc` *`fpc-slot`* `pic` *`number`*`]` hierarchy level. If you do not specify the tunnel bandwidth then, the tunnel interface can have a maximum bandwidth of up to 120 Gbps.

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth ;
        }
    }
}
```

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer.

In the following example, you create tunnel interfaces on PIC 1 of MPC 5 with 40 Gbps of bandwidth reserved for tunnel traffic. `fpc` *`slot-number`* is the slot number of the MPC. In this configuration, the tunnel interfaces created are gr-5/1/1, pe-5/1/1, pd-5/1/1, vt-5/1/1, and so on.

To create a 40-Gbps tunnel interface, use the following configuration:

```
[edit chassis]
fpc 5 {
    pic 1 {
    tunnel-services {
        bandwidth 40g;
    }
}
}
```

**SEE ALSO**

Tunnel Interfaces on MX Series Routers with Line Cards (MPC7E through MPC11E) | 6

# Configuring Tunnel Interfaces on MX Series Routers with MX2K-MPC8E

MX2K-MPC8E support a total of four inline tunnels per MPC, one per PIC. You can create a set of tunnel interfaces per PIC slot up to a maximum of four slots from 0 through 3.

To configure the tunnel interfaces, include the `tunnel-services` statement and an optional bandwidth of 1–120Gbps at the `[edit chassis fpc fpc-slot pic number ]` hierarchy level. If you do not specify the tunnel bandwidth then, the tunnel interface can have a maximum bandwidth of up to 120 Gbps.

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth;
        }
    }
}
```

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer.

In the following example, you create tunnel interfaces on PIC 1 of MPC 5 with 40 Gbps of bandwidth reserved for tunnel traffic. `fpc slot-number` is the slot number of the MPC. In this configuration, the tunnel interfaces created are gr-5/1/1, pe-5/1/1, pd-5/1/1, vt-5/1/1, and so on.

To create a 40-Gbps tunnel interface, use the following configuration:

```
[edit chassis]
fpc 5 {
    pic 1 {
    tunnel-services {
        bandwidth 40g;
    }
}
}
```

**SEE ALSO**

Tunnel Interfaces on MX Series Routers with Line Cards (MPC7E through MPC11E) | 6

## Configuring Tunnel Interfaces on MX Series Routers with MX2K-MPC9E

MX2K-MPC9E supports a total of four inline tunnels per MPC, one per PIC. You can create a set of tunnel interfaces per PIC slot up to a maximum of four slots from 0 through 3.

To configure the tunnel interfaces, include the `tunnel-services` statement and an optional bandwidth in the range 1–200Gbps at the `[edit chassis fpc` *`fpc-slot`* `pic` *`number`* `]` hierarchy level. If you do not specify the tunnel bandwidth then, the tunnel interface can have a maximum bandwidth of up to 200 Gbps.

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth ;
        }
    }
}
```

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer.

In the following example, you create tunnel interfaces on PIC 1 of MPC 5 with 40 Gbps of bandwidth reserved for tunnel traffic. `fpc` *`slot-number`* is the slot number of the MPC. In this configuration, the tunnel interfaces created are gr-5/1/1, pe-5/1/1, pd-5/1/1, vt-5/1/1, and so on.

To create a 40-Gbps tunnel interface, use the following configuration:

```
[edit chassis]
fpc 5 {
    pic 1 {
    tunnel-services {
        bandwidth 40g;
    }
}
}
```

#### SEE ALSO

Tunnel Interfaces on MX Series Routers with Line Cards (MPC7E through MPC11E) | 6

# Configuring Tunnel Interfaces on MX Series Routers with MPC10E-10C and MPC10E-15C

**SUMMARY**

MPC10E-10C supports two inline tunnel interfaces per MPC. You can create a set of tunnel interfaces per PIC slot up to a maximum of two slots- 0 and 1. MPC10E-10C MPC10E-15C supports three inline per PIC slot up to a maximum of two slots- 0 and 1. MPC10E-10C MPC10E-15C supports three inline tunnel interfaces per MPC. You can create a set of tunnel interfaces per PIC slot up to a maximum of three slots from 0 through 2.

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth ;
        }
    }
}
```

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer.

In the following example, you create tunnel interfaces on PIC 1 of MPC with 40 Gbps of bandwidth reserved for tunnel traffic. `fpc slot-number` is the slot number of the MPC. In this configuration, the tunnel interfaces created are gr-5/1/0, pe-5/1/0, pd-5/1/1, lt-5/1/0, and so on.

To create a 40-Gbps tunnel interface, use the following configuration:

```
[edit chassis]
fpc 5 {
    pic 1 {
    tunnel-services {
        bandwidth 40g;
    }
```

```
}
}
```

SEE ALSO

Tunnel Interfaces on MX Series Routers with Line Cards (MPC7E through MPC11E) | 6

## Configuring Tunnel Interfaces on MX Series Routers with MX2K-MPC11E

SUMMARY

MX2K-MPC11E supports a total of eight inline tunnels per MPC, one per PIC. You can create a set of tunnel interfaces per PIC slot up to a maximum of eight slots from 0 through 7.

To configure the tunnel interfaces, include the `tunnel-services` statement and an optional bandwidth in the range 1–400Gbps at the `[edit chassis fpc` *`fpc-slot`* `pic` *`number`* `]` hierarchy level. If you do not specify the tunnel bandwidth then, the tunnel interface can have a maximum bandwidth of up to 400 Gbps.

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth ;
        }
    }
}
```

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer.

In the following example, you create tunnel interfaces on PIC 1 of MPC 5 with 40 Gbps of bandwidth reserved for tunnel traffic. `fpc` *`slot-number`* is the slot number of the MPC. In this configuration, the tunnel interfaces created are gr-5/1/0, pe-5/1/0, pd-5/1/0, lt-5/1/0, and so on.

To create a 40-Gbps tunnel interface, use the following configuration:

```
[edit chassis]
fpc 5 {
    pic 1 {
    tunnel-services {
        bandwidth 40g;
    }
}
}
```

SEE ALSO

Tunnel Interfaces on MX Series Routers with Line Cards (MPC7E through MPC11E) | 6

## Configuring Tunnel Interfaces on MX Series Routers with MX10K-LC9600

MX10K-LC9600 MPC supports a total of six inline tunnels per MPC. Each tunnel PIC can have up to 4 tunnel interfaces.

To configure the tunnel interfaces, include the `tunnel-services` statement and bandwidth upto 400Gbps at the `[edit chassis fpc` *`fpc-slot`* `pic` *`number`*`]` hierarchy level. If you do not specify the tunnel bandwidth then, the tunnel interface can have a maximum bandwidth of up to 400 Gbps.

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
         tunnel-port {
            bandwidth ;
        }
    }
  }
}
```

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer.

In the following example, you create tunnel interfaces on PIC 1 of MX10K-LC9600 and tunnel port number 1 with 40 Gbps of bandwidth reserved for tunnel traffic. `fpc` *`slot-number`* is the slot number of the MPC. In this configuration, the tunnel interfaces created are gr-5/1/1, pe-5/1/1, pd-5/1/1, vt-5/1/1, and so on.

To create a 40-Gbps tunnel interface, use the following configuration:

```
[edit chassis]
fpc 5 {
    pic 1 {
    tunnel-services {
      tunnel-port 1 {
        bandwidth 40g;
    }
}
}
}
```

## Example: Configuring Tunnel Interfaces on a Gigabit Ethernet 40-Port DPC

The following example shows how to create tunnel interfaces on Packet Forwarding Engine 1 of DPC 4 with 1 Gbps of bandwidth reserved for tunnel services. On a Gigabit Ethernet 40-port DPC, tunnel interfaces coexist with Ethernet interfaces. With this configuration, the Gigabit Ethernet interfaces are **ge-4/1/0** through **ge-4/1/9**. The tunnel interfaces created are **gr-4/1/10**, **pe-4/1/10**, **pd-4/1/10**, **vt-4/1/10** and so on.

```
[edit chassis]
fpc 4 pic 1 {
    tunnel-services {
        bandwidth 1g;
    }
}
```

SEE ALSO

Configuring the Junos OS to Support ILMI for Cell Relay Encapsulation on an ATM2 IQ PIC

*bandwidth (Tunnel Services)*

*tunnel-services (Chassis)*

## Example: Configuring Tunnel Interfaces on a 10-Gigabit Ethernet 4-Port DPC

In this example, you create tunnel interfaces on Packet Forwarding Engine 0 of DPC 4 with 10 Gbps of bandwidth reserved for tunnel traffic. Ethernet and tunnel interfaces cannot coexist on the same Packet Forwarding Engine of a 10-Gigabit Ethernet 4-port DPC. With this configuration, the tunnel interfaces created are **gr-4/0/0**, **pe-4/0/0**, **pd-4/0/0**, **vt-4/0/0** and so on.

```
[edit chassis]
fpc 4 pic 0 {
    tunnel-services {
        bandwidth 10g;
    }
}
```

SEE ALSO

*bandwidth (Tunnel Services)*

*tunnel-services (Chassis)*

## Configuring Tunnel Interfaces on MX 204 Routers

The MX204 router is a fixed-configuration router, and supports one fixed Routing Engine. It has two PICs and contains a total of twelve fixed ports, in two groups of four and eight, respectively. The set of four ports (referred to as the PIC 0 ports) are rate selectable and can be configured at 10-Gbps (by using a breakout cable), 40-Gbps, or 100-Gbps speed. However, not all the ports support all the three speeds. The set of eight ports (referred to as PIC 1 ports) operate at a fixed speed of 10-Gbps.

The MX204 router supports two inline tunnels - one per PIC. To configure the tunnel interfaces, include the `tunnel-services` statement and an optional bandwidth of 1 Gbps through 200 Gbps at the `[edit chassis` fpc *fpc-slot* pic *number*] hierarchy level. If you do not specify the tunnel bandwidth then, the tunnel interface can have a maximum bandwidth of up to 200 Gbps.

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth ;
        }
    }
}
```

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the CLI Explorer.

In the following example, you create tunnel interfaces on PIC 0 of MPC 0 with 40 Gbps of bandwidth reserved for tunnel traffic. fpc *slot-number* is the slot number of the MPC.

To create a 40-Gbps tunnel interface, use the following configuration:

```
[edit chassis]
fpc 0 {
    pic 0 {
    tunnel-services {
        bandwidth 40g;
    }
}
}
```

## SEE ALSO

Tunnel Interfaces on MX Series Routers with Line Cards (MPC7E through MPC11E) | 6

## Configuring Tunnel Interfaces on T4000 Routers

To create tunnel interfaces on a T4000 Core Router, include the following statements at the `[edit chassis]` hierarchy level:

```
[edit chassis]
fpc slot-number {
    pic number {
        tunnel-services {
            bandwidth bandwidth-value;
        }
    }
}
```

`fpc` *`slot-number`* denotes the slot number of the FPC. On the T4000 router, the range is 0 through 7.

**NOTE**:

- This applies only to the T4000 Type 5 FPC. If any other type of FPC is configured in this slot, this configuration is ignored and no tunnel physical interface is created.
- When you use Type 5 FPCs, the tunnel interfaces are soft interfaces and allow as much traffic as the forwarding-path allows. So, it is advantageous to setup tunnel services without artificially limiting traffic by setting the `bandwidth` statement.

`pic` *`number`* on the T4000 router is 0 or 1.

`bandwidth` *bandwidth-value* is the amount of bandwidth to reserve for the tunnel traffic on each Packet Forwarding Engine. The bandwidth value accepted includes every multiple of 10g up to 100g.

If you specify a bandwidth that is not compatible, tunnel services are not activated. For example, you cannot specify a bandwidth of 1 Gbps for a Packet Forwarding Engine on a 100-Gigabit Ethernet PIC with CFP.

To verify that the tunnel interfaces have been created, issue the `show interfaces terse` operational mode command. For more information, see the *Junos Interfaces Command Reference*.

#### SEE ALSO

*bandwidth (Tunnel Services)*

*tunnel-services (Chassis)*

# Configuring Tunnel Interfaces on MX Series Routers and PTX Series Routers

IN THIS SECTION



## Understanding Tunnel Interfaces on MX Series Routers

- Interface (gr-, lt-, and ip-)-based tunnels —You can configure interface-based tunnels when the bandwidth profile enforcement is required. The GRE tunnels support IPv4, IPv6, MPLS, ISO, and Ethernet payload, whereas the IP-IP tunnels support IPv4 and IPv6 payload.

  You can configure interface-based tunnels to implement:

  - Tunneling over an IP network with bandwidth enforcement per tunnel. For example, toward the remote site.
  - Steering for DDoS cleaning.
  - Mirroring to remote destinations.

  During the GRE process, after the GRE header is added to the packet, the packet is looped back to the same Packet Forwarding Engine for a second lookup. The packet enters the ingress pipeline back through the loopback interface, which has a limited bandwidth of 400G. The encapsulated packet is then forwarded to the destination.

  GRE tunnel de-encapsulation is implemented by inline tunnel termination and does not use the loopback stream. However, the overall packet performance of the Packet Forwarding Engine is impacted due to extra fabric hops as the traffic flow switches from one Packet Forwarding Engine to another.

- Flexible tunnel interfaces (FTIs)—You can configure FTIs when the bandwidth profile enforcement is not required. FTIs support both IPv4 and IPv6 payload. You can configure FTIs to implement:

  - Mirroring to remote destinations.

- IP fabrics with IP-IP overlays.
- Steering for DDoS cleaning.

The packet performance is reduced due to the extra lookup after encapsulation and de-encapsulation.

- Dynamic tunnels—You can configure dynamic tunnels to design data center gateways.

  In the dynamic tunnels implementation with loopback avoidance, the packet performance is reduced due to the extra lookup after an encapsulation or de-encapsulation.

- Firewall filter-based tunnels— These tunnels support:
  - GRE and GRE in UDP encapsulation.
  - GRE, IP-IP, and GRE in UDP de-encapsulation.

  You can configure firewall filter-based tunnels to design data center gateways.

  You can configure GRE-based encapsualtion and de-encapsulation using a firewall filter action without using a tunnel interface. Encapsulation and de-encapsulation happens at the Packet Forwarding Engine which processes the filter. MX Series routers support firewall filters at:

  - Interface level on input (executed on the ingress Packet Forwarding Engine)
  - Interface level on output (executed on the egress Packet Forwarding Engine)
  - Forwarding table level (either before route lookup or after route lookup). In both cases, the filter is executed on the ingress PFE

  In this scenario, the packet performance is reduced due to the extra lookup after encapsulation. In case of de-encapsulation, the packet performance is reduced due to the filter configuration.

## Understanding Tunnel Interfaces on PTX Series Routers

**IN THIS SECTION**

You can configuring tunnel interfaces to implement different features on the PTX Series routers. The following sections provide an overview of the features implemented on different PTX Series routers.

## Tunnel Interfaces on PTX10001-36MR, PTX10004, PTX10008, and PTX10016-Overview

This section provides information about configuring tunnel interfaces to implement different features on PTX10001-36MR, PTX10004, PTX10008, and PTX10016 routers with Junos OS Evolved.

- Flexible tunnel interfaces (FTIs)—You can configure FTI-based tunnels to implement:
  - Steering for DDoS cleaning.
  - De-encapsulation for all dynamic tunnels use cases.

  These tunnels support GRE, UDP, and IP-IP encapsulation and de-encapsulation options. The reduction in the packet performance depends on the encapsulation option. Encapsulation supports flattened next-hop topology..

  You can configure GRE tunnels on flexible tunnel interfaces. When you enable the `tunnel-termination` statement at the `[edit interfaces fti0 unit` *`unit-number`* `family (inet | inet6)]` hierearchy, tunnels are terminated on the WAN interface before any other actions, such as sampling, port mirroring, or filtering, are applied.

- Flexible tunnel interfaces (FTIs) through a loopback—You can configure FTIs to implement mirroring to remote destinations.

  On an FTI, after tunnel encapsulation, traffic is sent into the loopback interface (on the ingress Packet Forwarding Engine) and later to the ultimate destination. The destinations could include those behind segment routing–traffic engineered (SR-TE) next hops. The loopback interface has a limited bandwidth of 400G.

  You can configure GRE/IP-in-IP/UDP tunnel encapsulation on FTIs using the loopback interface. You can configure encapsulation by using the command `tunnel encapsulation (gre|ipip|udp) source` *`address`* `destination` *`address`* at the `[edit interfaces fti0 unit` *`unit-number`* hierarchy. You must consider the following points while configuring this feature:

  - Adding `tunnel-termination` makes the tunnel decap-only tunnel and encapsulation is disabled.
  - Specifying both the source and destination address is mandatory when you do not configure the command.
  - Configuring a variable prefix mask on the source address is not allowed

- Dynamic tunnels— You can configure dynamic tunnels to design:

  - IP fabrics.
  - IP overlays.
  - Data center gateways.

  These tunnels support IP-IP encapsulation.

  PTX Series routers with Junos OS Evolved, do not support dynamic tunnels for de-encapsulation.. Instead, you can use static FTI tunnels for de-encapsulation, without specifying the destination address. The tunnels are configured with the de-encapsulation-only option.

  You can configure next-hop-based dynamic UDP tunnels, also known as MPLS-over-UDP tunnels. Junos OS dynamically creates next hops to resolve the tunnel destination route. You can also use policy control to resolve MPLS-over-UDP tunnels over select IP prefixes. Because when the next-hops are enabled by default, the MPLS-over-UDP feature provides a scaling advantage for the number of IP tunnels supported on the router.

## Tunnel Interfaces on PTX1000, PTX5000, and PTX10002-50C-Overview

This section provides information about configuring tunnel interfaces to implement different features on PTX1000, PTX5000, and PTX10002-50C routers with Junos OS Evolved.

- Flexible tunnel interfaces (FTIs)—You can configure FTIs to implement IP fabric. We prefer this option when we need to reach tunnel destinations through IP. These tunnels support:
  - GRE, UDP, and IP-IP encapsulation options.
  - UDP and IP-IP de-encapsulation options.

  The packet performance is reduced due to the extra lookup after encapsulation and de-encapsulation.

- Dynamic tunnels—You can configure dynamic tunnels to design data center gateway. The packet performance is reduced due to extra lookup after encapsulation and de-encapsulation.

- Firewall filter-based tunnels—You can configure firewall filter-based tunnels to implement egress peering engineering (EPE). The packet performance is reduced during encapsulation due to the extra lookup.

- Mirroring to remote destination— You can implement tunneling of mirrored packets to remote destinations. The packet performance is reduced during encapsulation due to the extra lookup.

# Use Cases Implemented by Configuring Tunnels on MX Series and PTX Series Routers

IN THIS SECTION



This section provides information about some of the use cases of configuring tunnel interfaces to implement different features (use cases) on the MX series and PTX series routers.

## Port Mirroring to Remote Destinations

You can use port mirroring for traffic analysis on routers and switches that, unlike hubs, do not broadcast packets to every port on the destination device. Port mirroring sends copies of all packets or policy-based sample packets to local or remote analyzers where you can monitor and analyze the data.

For an MX Series router configured as a provider edge (PE) router on the customer-facing edge of a service provider network, you can apply a Layer 2 port-mirroring firewall filter at the ingress and egress points to mirror the traffic between the MX Series router and customer edge (CE) devices, such as routers and Ethernet switches.

On MX Series routers, you can mirror traffic arriving at tunnel interfaces to multiple destinations. You specify two or more destinations in a next-hop group, define a firewall filter that references the next-hop group as the filter action, and then apply the filter to a logical tunnel interface (`lt-`) or virtual tunnel interfaces (`vt-`) on the MX Series router.

See Configuring Port Mirroring and Configuring Port Mirroring for Remote Destinations.

When the data path traverses a flexible tunnel interface (FTI)-based tunnel, the router sends the output packet with tunnel encapsulation. You can set up a configuration that mirrors the original packet as well as the packet with all encapsulations as it as it exits the interface out.

To enable mirroring based on a filter installed on the FTI:

- You mark packets for mirroring using the policy action at the FTI. The router typically uses the policy action to select the egress rewrite rule, but in this case, it uses the policy action to mark interesting packets with an internal policy attribute, without any special rewrite rule configured.

- You have the software intercept packets that match the specific policy on the egress WAN side and initiate the `l2-mirror` action. Packets are reported with Layer 2 header information, including tunnel encapsulation.

  See No Link Title and No Link Title

## Data Center Gateways

Data center gateways interconnect the Internet or enterprise VPNs on one side and virtual machines hosted on servers on the other side. Overlay transport technologies such as MPLS-over-GRE or MPLS-over-UDP are part of a data center design. Host routes are communicated to the data center gateway from the SDN controller and imported into virtual routing and forwarding (VRF) or Internet routing contexts.. The data center servers are reachable through next-hop-based dynamic tunnels. Tunnels are established on servers in the BGP route protocol next-hop resolution process.

As described in, *RFC 5549*, IPv4 traffic is tunneled from CPE devices to IPv4-over-IPv6 gateways. These gateways are announced to CPE devices through anycast addresses. The gateway devices then create dynamic IPv4-over-IPv6 tunnels to remote CPE devices and advertise IPv4 aggregate routes to steer traffic. Route reflectors with programmable interfaces inject the tunnel information into the network. The route reflectors are connected through internal BGP (IBGP) to gateway routers, which advertise the IPv4 addresses of host routes with IPv6 addresses as the next hop.

The MPLS-over-UDP tunnel is handled as follows:

1. After an MPLS-over-UDP tunnel is configured, a tunnel destination mask route with a tunnel composite next hop is created for the tunnel in the inet.3 routing table. This IP tunnel route is withdrawn only when the dynamic tunnel configuration is deleted.

   The tunnel composite next-hop attributes include the following:

   - When the Layer 3 VPN composite next hop is disabled—Source and destination address, encapsulation string, and VPN label.

   - When the Layer 3 VPN composite next hop and per-prefix VPN label allocation are enabled—Source address, destination address, and encapsulation string.

   - When the Layer 3 VPN composite next hop is enabled and the per-prefix VPN label allocation is disabled—Source address, destination address, and encapsulation string. The route in this case is added to the other VRF instance table with a secondary route.

2. The provider edge (PE) devices are interconnected using an IBGP session. The IBGP route next hop to a remote BGP neighbor is the protocol next hop, which is resolved using the tunnel mask route with the tunnel next hop.

3. After the protocol next hop is resolved over the tunnel composite next hop, indirect next hops with forwarding next hops are created.

4. The tunnel composite next hop is used to forward the next hops of the indirect next hops.

See Example: Configuring Next-Hop-Based MPLS-Over-UDP Dynamic Tunnels and Example: Configuring Next-Hop-Based IP-Over-IP Dynamic Tunnels

# Configuring Flexible Tunnel Interfaces

IN THIS SECTION



## Flexible Tunnel Interfaces Overview

IN THIS SECTION



A flexible tunnel interface (FTI) is a type of logical tunnel interface that uses static routing and BGP protocols to exchange routes over a tunnel that connects endpoints to routers.

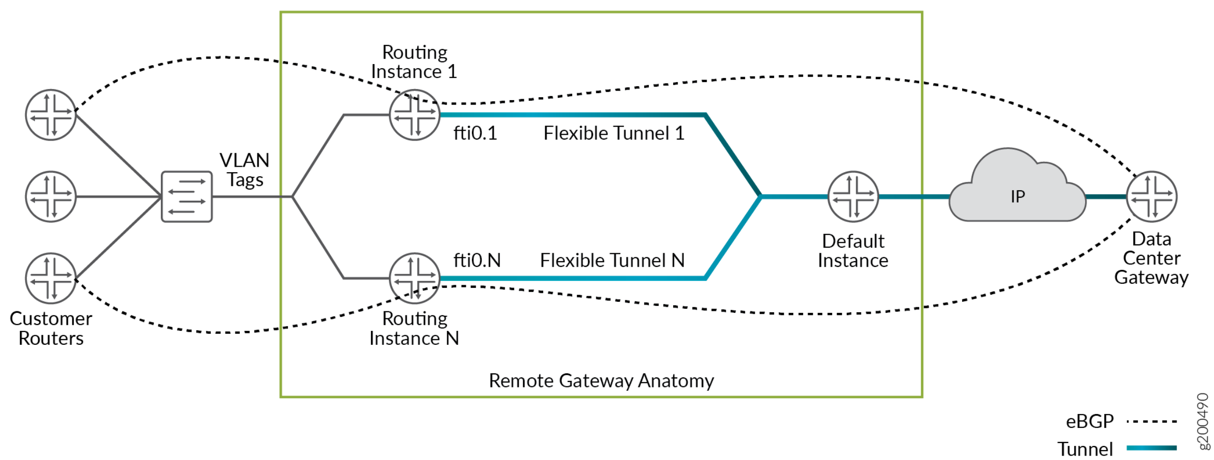
### Flexible Tunnel Interfaces on MX Series Routers

On MX Series routers, FTIs have the following features:

- FTI supports only VXLAN encapsulation with Layer 2 pseudo-headers.
- FTI is used between a router and a server hosting multiple virtual machines, or between routers in two different data centers.
- FTI can be configured as port-mirror destinations.
- FTI support logical interface statistics streaming.

In the VXLAN encapsulation process, the Layer 2 address is populated with "pseudo" source (source MAC: 00-00-5E-00-52-00) and destination (destination MAC: 00-00-5E-00-52-01) MAC addresses without VLAN tagging; however, these addresses are ignored when the packets reach the remote endpoint. The remote endpoint is identified by the destination IP address and a specified destination UDP port number. The corresponding FTI on the remote endpoint is identified by the virtual network identifier (VNI) value, the source IP address of the tunnel, and the destination UDP port number. All of these values can be configured on an FTI with VXLAN encapsulation.

**Figure 1: FTIs Connecting Remote Devices to a Virtual Private Cloud**




[Figure 1 on page 39](#) illustrates how an FTI works to provide connectivity into a virtual private cloud from a remote location. Individual flexible tunnels (1 through *N*) are provisioned for every customer. The customer-facing logical interface and the corresponding FTIs are configured to operate in one routing instance. The FTI uses BGP protocols (eBGP and iBGP) to carry packets from the customer device to the remote gateway and vice versa.

## Flexible Tunnel Interfaces on PTX Series Routers and QFX Series Switches

On PTX Series routers and QFX Series switches, FTIs have the following features:

- FTI is supported in releases starting Junos OS Release 19.3R1.

- FTI supports only UDP encapsulation.
- FTI can be initiated at any place in the MPLS tunnel: MPLS transit, ingress, egress, and PHP.
- FTI with UDP encapsulation supports the following payloads:
  - IPV4 inside IPV4 UDP packet
  - IPV6 inside IPV4 UDP packet
  - MPLS inside IPV4 UDP packet
  - ISO inside IPV4 UDP packet

FTI with UDP encapsulation supports the following features and functionality:

- MPLS link protection and node-link protection.
- Manual configuration of RSVP bandwidth.
- BFD support for liveliness detection, excluding BFD over LDP and RSVP.
- Support for the following protocols:
  - BGP
  - RSVP
  - LDP
  - OSPF
  - ISIS
- Static routes.
- FTI logical interface statistics.
- MTU configuration on FTI and fragmentation of payload before entering the tunnel.
- Underlay can be Aggregated Ethernet or regular interface, and can be tagged sub-interface or regular Layer 3 interfaces.
- Overlay and underlay ECMP.

To configure an FTI interface with UDP encapsulation, include the "udp" on page 246 statement at the `[edit interfaces` *`fti0`* `unit` *`unit`* `tunnel encapsulation]` hierarchy level.

For example:

```
[edit interfaces]
fti0 {
    unit 0-1000 {
        tunnel {
            encapsulation udp {
                source {
                    address ipv4_address;
                }
                destination{
                    address ipv4_address;
                }
            }
        }
        family inet {
            destination-udp-port udp port [range 10000-10009] ;
        }
        family inet6 {
            destination-udp-port udp port [range 10010-10019];
        }
        family mpls {
            destination-udp-port udp port [range 10020-10029];
        }
        family iso {
            destination-udp-port udp port [range 10030-10039];
        }
    }
}
```

## MPLS Support for FTI tunnels on PTX Series Routers

Starting In Junos OS Evolved Release 21.4R1, you can configure MPLS protocols over FTI tunnels, thereby transporting MPLS packets over IP networks which does not support MPLS.

In Junos OS Evolved Release 21.4R1, generic routing encapsulation (GRE) and UDP tunnels support MPLS protocol for IPv4 and IPv6 traffic. You can configure encapsulation and decapsulation for the GRE and UDP tunnels.

The following features are supported :

- Encapsulation and decapsulation for IPv4 and IPv6 traffic

- UDP port number configuration
- MPLS node-link protection
- Ingress, egress, PHP, and transit roles for LSP
- Ping and traceroute support in ingress, egress, PHP, and transit roles for LSP
- Overlay and underlay ECMP
- Manual configuration of RSVP bandwidth.
- MPLS services
  - L3VPN
  - 6VPE
  - L2 circuit
  - BGP-LU with per nexhop or prefix label
- Routing instance
- Class-of-service (CoS) including the configuration of rewrite rules and classifiers
- MTU configuration and fragmentation of payload
- BFD support for liveliness detection.
- Jvision

The following features and functionality are not supported:

- MPLS link protection
- RSVP bandwidth Inheritance based on next hop to tunnel destination for FTI interfaces
- TTL propagation.
- Class-of-service on tunnel endpoints .
- FT-over-FT resolution .
- FT destination IP should be reachable through IGP and not BGP (no indirect next hop). The reachability should be through an IPV4 route and not through an LSP.
- Path MTU discovery .

To allow the MPLS traffic on the UDP tunnels include the `mpls` *port-number* statement at the `[edit forwarding-options tunnels udp port-profile` *profile-name*`]` hierarchy level. To allow the MPLS traffic on the GRE tunnels, include the `mpls` statement at the `[edit interfaces` *fti0* `unit` *unit* `family]` hierarchy.

For example:

```
[edit forwarding-options]
    tunnels {
udp {
       port-profile  p1 {
              inet  <port num>
              inet6  <port num>
              mpls  <port num>
             iso <port num>
}
}
}
```

## Benefits of Flexible Tunnel Interfaces

- Entropy and load balancing occur in transit. Unlike over tunnel encapsulations, such as IP in IP or generic routing encapsulation (GRE), VXLAN encapsulation supports passing of the hash computation result in the source port of the UDP datagram. This enables you to load-balance traffic efficiently in transit.
- FTIs have an extensible design that enables them to support multiple encapsulations.
- The `vni` attribute of the VXLAN encapsulation in FTIs helps in customer isolation.
- FTIs with UDP encapsulation use the source and destination port in the UDP header. Because the UDP source port is derived from the hash value of the inner payload, you can benefit from better traffic distribution over ECMP.

## Limitations of Flexible Tunnel Interfaces

- Policing follows the distributed forwarding model of the FTIs; therefore provisioned bandwidth limits are enforced at an individual Packet Forwarding Engine level. As a result, more traffic might be admitted.
- Currently, FTI-tunneled traffic is strictly routed in the `inet.0` instance. Therefore, FTIs support only IPv4 traffic.
- The MX80 does not support FTIs.

- Class-of-service (CoS) configuration, including the configuration of rewrite rules and classifiers is not supported on FTIs.
- Time-to-live (TTL) on the tunnel header is set to the default value 100.
- Differentiated Services code point (DSCP) value is set to the default value 0, but internal forwarding class and loss priority fields are retained and can be used to rewrite DSCP in the egress interface rewrite rules.
- IP fragmentation is not supported on FTIs.

FTI with UDP encapsulation do not support the following features and functionality:

- BFD over LDP and RSVP is not supported.
- Aggregate Ethernet member statistics on QFX1000 device is not supported.
- 10,000 routes per FTI logical interface is not supported.
- Routing instance is not supported.
- Logical systems is not supported.
- Path MTU discovery is not supported.
- Policing and firewall is not supported.
- BGP signaling for UDP tunnels is not supported.
- Class-of-service on tunnel endpoints is not supported.
- TTL propagation is not supported.
- Multicast traffic is not supported.
- Plain IPV6 UDP tunnel is not supported.
- Anti-spoofing check for tunneled traffic is not supported.
- MPLS FRR is not supported.
- FT-over-FT resolution is not supported.
- FT destination IP should be reachable through IGP and not BGP (no indirect next hop). The reachability should be through an IPV4 route and not through an LSP.
- FT physical interface level statistics is not supported.
- All the interfaces under FTI except for fti0 are not supported.

- Un-numbered address is not supported.

SEE ALSO

show interfaces fti | 358
vxlan-gpe (FTI) | 253
vni(Interfaces) | 252
destination-udp-port (FTI) | 186

## Configuring Flexible Tunnel Interfaces

IN THIS SECTION



You can configure flexible tunnel Interfaces (FTIs) that support the Virtual Extensible LAN (VXLAN) encapsulation with Layer 2 pseudo-headers on MX Series routers, or UDP encapsulation on PTX Series routers and QFX Series switches. A flexible tunnel interface (FTI) is a point-to-point Layer 3 interface that can be used to create IPv4 and IPv6 overlays over an IPv4 transport network. A BGP protocol session can be configured to run over FTIs in order to distribute routing information.

The following sections describe how to configure FTIs on your device and to enable multiple encapsulations using the `udp` or `vxlan-`*gpe* parameter under the mandatory `tunnel-endpoint` *vxlan* encapsulation identified with the `vni` and `destination-udp-port` values:

### Configuring FTI on PE1

You can configure an FTI by including the `tunnel-endpoint vxlan` statement at the `[edit interfaces]` hierarchy level.

To configure an FTI and define its attributes for an IPv4 network:

1. In configuration mode, go to the `[edit interfaces]` hierarchy level.

```
[edit]
user@host# edit interfaces
```

2. On MX Series routers, configure a logical unit for the interface and the encapsulation `vxlan-gpe`. The unit is a logical interface configured on the physical device. Specify the value of the unit from 0 through 8191. VXLAN is defined as an encapsulation format that encapsulates Ethernet frames in an outer UDP/IP transport.

   **NOTE**: The capabilities of VXLAN-GPE are a super-set of what VXLAN without protocol extension offers. Therefore generic vxlan-gpe hierarchy is introduced to configure VXLAN tunnel encapsulation attributes; however, only regular VXLAN encapsulation without protocol extensions and pseudo Layer 2 MAC is used. The pseudo Layer 2 address is populated with "pseudo" source (source MAC: 00-00-5E-00-52-00) and destination (destination MAC: 00-00-5E-00-52-01) MAC addresses without VLAN tagging.

   On MX Series routers:

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number tunnel encapsulation vxlan-gpe
```

   Starting in Junos OS Release 19.3R1, you can configure flexible tunnel interfaces (FTIs) with UDP encapsulation on the PTX Series routers and the QFX Series switches, which provide support for static UDP tunnels only.

   FTIs with UDP encapsulation provides the benefit of better traffic distribution over ECMP, that is achieved by the UDP source port derived from the hash value of the inner payload. In addition to this, the other benefits of this feature include, shortened interface hop counts, smooth IGP domain separation, and reduced operational complexity.

   On PTX Series routers and QFX Series switches:

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number tunnel encapsulation udp
```

3. Configure the source address for the interface. The source address is the IPv4 address or address range of the encapsulator (the local ingress PE router).

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number tunnel encapsulation vxlan-gpe source
address
```

**NOTE**: The source address can be a global WAN address and loopback address (`lo0`) is not mandatory.

4. Configure the destination address for the interface. The destination address is the IPv4 address of the tunnel endpoint destination.

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number tunnel encapsulation vxlan-gpe
destination-address address
```

5. Configure `tunnel-endpoint` with the encapsulation `vxlan`. This step is mandatory to enable Layer 2 pseudo-header with VXLAN encapsulation.

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number tunnel encapsulation vxlan-gpe tunnel-
endpoint vxlan
```

6. Specify the UDP port value of the destination to be used in the UDP header for the generated frames. The numeric value for `destination-udp-port` identifies the endpoint. Specify the value of `destination-udp-port` from 1 through 65,535.

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number tunnel encapsulation vxlan-gpe
destination-udp-port port-number
```

7. Specify the virtual network identifier (VNI) value to be used to identify the encapsulation, `vxlan-gpe`. Specify the value of the `vni` from 0 through 16,777,214.

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number tunnel encapsulation vxlan-gpe vni vni-number
```

8. Configure an IPv4 address of an interface (signified by family `inet`). For IPv6 address configuration, use the `inet6` family.

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number family inet address address
```

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

## Verification

**IN THIS SECTION**



### Purpose

Verify that the FTI is configured and verify its status.

### Action

In configuration mode, you can verify if FTI on MX Series router has been configured by executing the `show interfaces fti` *`number`* command.

```
user@host# show interfaces fti0
Physical interface: fti0, Enabled, Physical link is Up
  Interface index: 136, SNMP ifIndex: 504
  Type: FTI, Link-level type: Flexible-tunnel-Interface, MTU: Unlimited, Speed: Unlimited
  Device flags   : Present Running
```

```
Interface flags: SNMP-Traps
Link type      : Full-Duplex
Link flags     : None
Last flapped   : Never
  Input packets : 0
  Output packets: 0

Logical interface fti0.0 (Index 340) (SNMP ifIndex 581)
  Flags: Up Point-To-Point SNMP-Traps Encapsulation: VXLAN-GPEv4
  Destination UDP port: 4789, VNI: 1000, Source address: 5.5.5.5, Destination address: 6.6.6.6
  Input packets : 0
  Output packets: 0
  Protocol inet, MTU: Unlimited
  Max nh cache: 0, New hold nh limit: 0, Curr nh cnt: 0, Curr new hold cnt: 0, NH drop cnt: 0
    Flags: Sendbcast-pkt-to-re
    Addresses, Flags: Is-Preferred Is-Primary
      Destination: 3.3.3/24, Local: 3.3.3.5, Broadcast: 3.3.3.255
```

Similarly you can execute the `show interfaces fti0 detail`, `show interfaces fti0 extensive`, `show interfaces fti0 terse`, and `show interfaces fti0 statistics` commands to get more details FTIs. See "show interfaces fti" on page 358.

#### Meaning

The `show interfaces fti0` command displays the status of the FTIs that have been configured with the new encapsulation `vxlan-gpe`. The output verifies that the FTI is configured and the physical link is `up`.

### RELATED DOCUMENTATION


show interfaces fti | 358

vxlan-gpe (FTI) | 253

vni(Interfaces) | 252

destination-udp-port (FTI) | 186

# Example: Configuring Flexible Tunnel Interfaces on MX Series Routers

IN THIS SECTION



## Requirements

This example uses the following hardware and software components:

- An MX10003 and an MX Series 5G Universal Routing Platform.
- Junos OS Release 18.3 or later.

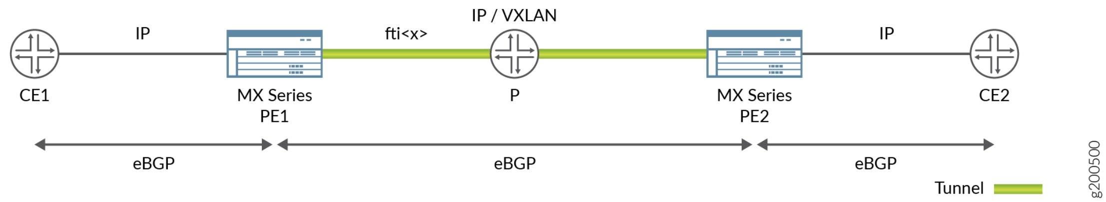
## Overview

In this example, flexible tunnel interfaces are used to create a Layer 3 VPN overlay network between two routers. In the actual deployment, one of the endpoints can be the server in a data center or a data center gateway.

Consider a sample topology in which a gateway device, PE1, functions as a link between the enterprise customers to represent the customer side for an FTI tunnel. eBGP is used to distribute routes between customer edge (CE1) and provider edge (PE1) devices. IPv4 is used for transmission of test frames over the Layer 3 network. This test is used to transfer the traffic between CE1 and CE2. Logical interfaces on both the routers are configured with IPv4 addresses to create an FTI to transfer the traffic of network devices for the IPv4 service.

Figure 2 on page 51 shows the sample topology of how an FTI performs for a Layer 3 IPv4 service.

**Figure 2: Flexible Tunnel Interfaces Topology**




## Configuration

**IN THIS SECTION**



In this example, you configure FTI for a Layer 3 IPv4 service that is between interface fti0 on PE1 and interface fti0 on PE2 to form a tunnel interface of the interconnecting routers.

### CLI Quick Configuration

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

#### To Configure Parameters on PE1

```
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe source address 198.51.100.1
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe destination address 198.51.100.2
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe tunnel-endpoint vxlan
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe destination-udp-port 4789
```

```
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe vni 22701
set interfaces fti0 unit 0 family inet address 198.51.100.1/24
```

To Configure Parameters on PE2

```
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe source address 198.51.100.2
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe destination address 198.51.100.1
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe tunnel-endpoint vxlan
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe destination-udp-port 4789
set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe vni 22701
set interfaces fti0 unit 0 family inet address 198.51.100.2/24
```

### Configuring on PE1

## Step-by-Step Procedure

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

To configure the parameters on PE1:

1. In configuration mode, go to the `[edit interfaces]` hierarchy level:

```
[edit]
user@host# edit interfaces
```

2. Configure the FTI and a logical unit and specify the protocol family.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe
```

3. Specify the source address for the logical interface.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe source address
198.51.100.1
```

4. Specify the destination address for the logical interface.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe destination address
198.51.100.2
```

5. Set `tunnel-endpoint` with the encapsulation `vxlan`.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe tunnel-endpoint vxlan
```

6. Specify the UDP port value of the destination to be used in the UDP header for the generated frames.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe destination-udp-port
4789
```

7. Specify the `vni` value to be used to identify the encapsulation `vxlan-gpe` on the interface.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe vni 22701
```

8. Specify the address type family for the interface.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 family inet address 198.51.100.1/24
```

#### Configuring on PE2

### Step-by-Step Procedure

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

To configure the parameters on PE2:

1. In configuration mode, go to the `[edit interfaces]` hierarchy level:

```
[edit]
user@host# edit interfaces
```

2. Configure the FTI and a logical unit and specify the protocol family.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe
```

3. Specify the source address for the logical interface.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe source address
198.51.100.2
```

4. Specify the destination address for the logical interface.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe destination address
198.51.100.1
```

5. Set `tunnel-endpoint` with the encapsulation `vxlan`.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe tunnel-endpoint vxlan
```

6. Specify the UDP port value of the destination to be used in the UDP header for the generated frames.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe destination-udp-port
4789
```

7. Specify the `vni` value to be used to identify the encapsulation `vxlan-gpe` on the interface.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 tunnel encapsulation vxlan-gpe vni 22701
```

8. Specify the address type family for the interface.

```
[set interfaces]
user@host# set interfaces fti0 unit 0 family inet address 198.51.100.2/24
```

After the configuration is successfully completed, you can view the parameters by entering the `show fti0` command.

#### Results

In configuration mode, confirm your configuration on PE1 and PE2 by entering the `show` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

Parameters on PE1:

```
[edit interfaces]
 fti0{
      unit 0 {
tunnel {
        encapsulation vxlan-gpe {
            source {
                address 198.51.100.1;
            }
            destination {
                address 198.51.100.2;
            }
            tunnel-endpoint vxlan;
```

```
            destination-udp-port 4789;
            vni 22701;
        }
    }
    family inet {
        address 198.51.100.1/24;
    }
}
```

Parameters on PE2:

```
[edit interfaces]
 fti0{
      unit 0 {
tunnel {
        encapsulation vxlan-gpe {
            source {
                address 198.51.100.2;
            }
            destination {
                address 198.51.100.1;
            }
            tunnel-endpoint vxlan;
            destination-udp-port 4789;
            vni 22701;
        }
    }
    family inet {
        address 198.51.100.2/24;
    }
}
```

After you have configured the interface, enter the `commit` command in configuration mode.

## Verification

**IN THIS SECTION**

#### Verifying the Results

##### Purpose

Verify that the necessary and desired tunnel displays the values configured for the FTI test that is run on the flexible tunnel between PE1 and PE2.

##### Action

In operational mode, enter the `show interfaces fti0` command to display status of the FTIs that have been configured with the new encapsulation `vxlan-gpe`. The output verifies that the FTI is configured and the physical link is `up`.

SEE ALSO

show interfaces fti | 358
vni(Interfaces) | 252

## Configuring IP-IP Decapsulation by Tunnel Termination on FTI

In filter based decapsulation, the decapsulated packets are re-circulated for inner header lookup and forwarded accordingly. However, tunnel termination is completed in a single pass of packet processing, thus providing performance improvement over filter based process. Starting in Junos OS Evolved Release 20.1R2, you can configure IP-IP decapsulation on a flexible tunnel interface on PTX series routers by configuring tunnel termination. You can configure IP-IP decapsulation on a flexible tunnel interface by configuring tunnel termination at the `[edit interfaces fti0 unit` *`number`* `tunnel encapsulation IPIP]` hierarchy level.

> **NOTE**: For the Junos OS Evolved Release 20.1R2, FTI does not support encapsulation.

To configure IP-IP decapsulation by tunnel termination:

1. On PTX Series routers, configure the FTI, logical unit for the interface, and the encapsulation `IPIP`. The unit is a logical interface configured on the physical device. Specify the value of the unit from 0 through 4096.

```
[edit interfaces]
user@host# set fti0 unit logical-unit-number tunnel encapsulation IPIP
```

2. For IP-IP decapsulation, configure the tunnel termination and specify the address family. For IPv6 address configuration, use the `inet6` family.

**NOTE**: For the Junos OS Evolved Release 20.1R2, this step is mandatory.

```
[edit interfaces fti0 unit logical-unit-number tunnel encapsulation IPIP]
user@host# set tunnel-termination
user@host# set famlly inet
```

3. Configure the source address and destination address for the interface.

```
[edit interfaces fti0 unit logical-unit-number tunnel encapsulation IPIP]
user@host# set source address 192.168.1.1
user@host# set destination address 192.168.2.1
```

4. Configure the routing instance for the FTI to facilitate routing table lookups. Create a virtual-router instance and associate the interface.

```
[edit routing-instances routing-instance-name]
user@host# set instance-type virtual-router
```

5. Verify the tunnel termination.

```
[edit]
user@host# show interfaces fti0 detail
```

**Release History Table**

| Release | Description |
|---|---|
| 20.1R2 Evo | Starting in Junos OS Evolved Release 20.1R2, you can configure IP-IP decapsulation on a flexible tunnel interface on PTX series routers by configuring tunnel termination. |
| 19.3R1 | FTI is supported in releases starting Junos OS Release 19.3R1. |
| 19.3R1 | Starting in Junos OS Release 19.3R1, you can configure flexible tunnel interfaces (FTIs) with UDP encapsulation on the PTX Series routers and the QFX Series switches, which provide support for static UDP tunnels only. |

# Configuring GRE Tunnel Interfaces

**IN THIS SECTION**



## Understanding Generic Routing Encapsulation

**IN THIS SECTION**

Generic routing encapsulation (GRE) provides a private, secure path for transporting packets through an otherwise public network by encapsulating (or tunneling) the packets.

This topic describes:

## Overview of GRE

GRE encapsulates data packets and redirects them to a device that de-encapsulates them and routes them to their final destination. This allows the source and destination routers to operate as if they have a virtual point-to-point connection with each other (because the outer header applied by GRE is transparent to the encapsulated payload packet). For example, GRE tunnels allow routing protocols such as RIP and OSPF to forward data packets from one router to another router across the Internet. In addition, GRE tunnels can encapsulate multicast data streams for transmission over the Internet.

GRE is described in RFC 2784 (obsoletes earlier RFCs 1701 and 1702). The routers support RFC 2784, but not completely. (For a list of limitations, see "Configuration Limitations" on page 61.)

As a *tunnel source router*, the router encapsulates a payload packet for transport through the tunnel to a destination network. The payload packet is first encapsulated in a GRE packet, and then the GRE packet is encapsulated in a delivery protocol. The router performing the role of a *tunnel remote router* extracts the tunneled packet and forwards the packet to its destination.

**NOTE**: Service chaining for GRE, NAT, and IPSec services on ACX1100-AC and ACX500 routers is not supported.

**NOTE**: Layer 2 over GRE is not supported in ACX2200 router.

ACX routers support OSPF routing protocol when a GRE tunnel is configured on a WAN interface.

## GRE Tunneling

Data is routed by the system to the GRE endpoint over routes established in the route table. (These routes can be statically configured or dynamically learned by routing protocols such as RIP or OSPF.) When a data packet is received by the GRE endpoint, it is de-encapsulated and routed again to its destination address.

GRE tunnels are *stateless*--that is, the endpoint of the tunnel contains no information about the state or availability of the remote tunnel endpoint. Therefore, the router operating as a tunnel source router cannot change the state of the GRE tunnel interface to down if the remote endpoint is unreachable.

For details about GRE tunneling, see:

### Encapsulation and De-Encapsulation on the Router

Encapsulation—A router operating as a tunnel source router encapsulates and forwards GRE packets as follows:

1. When a router receives a data packet (payload) to be tunneled, it sends the packet to the tunnel interface.
2. The tunnel interface encapsulates the data in a GRE packet and adds an outer IP header.
3. The IP packet is forwarded on the basis of the destination address in the outer IP header.

De-encapsulation—A router operating as a tunnel remote router handles GRE packets as follows:

1. When the destination router receives the IP packet from the tunnel interface, the outer IP header and GRE header are removed.
2. The packet is routed based on the inner IP header.

### Number of Source and Destination Tunnels Allowed on a Router

ACX routers support as many as 64 GRE tunnels between routers transmitting IPv4 or IPv6 payload packets over GRE.

## Configuration Limitations

Some GRE tunneling features are not currently available on ACX Series routers. Be aware of the following limitations when you are configuring GRE on an ACX router:

- Unsupported features—GRE on the ACX routers *does not support* the following features:
  - Virtual routing over GRE
  - Bidirectional Forwarding Detection (BFD) protocol over GRE distributed mode
  - MPLS over GRE tunnels
  - GRE keepalives
  - GRE keys, payload packet fragmentation, and sequence numbers for fragmented packets
  - BGP dynamic tunnels
  - RFC 1701 and RFC 1702
  - RFC 2890—Key and sequence number extensions to GRE
  - IPv6 as delivery header

    - GRE path MTU discovery
    - Load balancing when NNI is ECMP
    - Interface statistics on GRE interfaces
    - Class of service and firewall on GRE tunnel

- Routing Protocol—ACX routers do not support routing protocols on GRE interfaces. You need to disable routing on GRE interfaces under the [edit protocols] hierarchy. For example,

```
[edit]
user@host# show protocols
ospf {
    area 0.0.0.0 {
        interface all;
        interface gr-0/0/10.0 {
            disable;
        }
    }
}
```

**NOTE**: This limitation is applicable for all routing protocols (such as OSPF, ISIS).

SEE ALSO

Configuring Unicast Tunnels | 74

## Configuring Generic Routing Encapsulation Tunneling

IN THIS SECTION

Tunneling provides a private, secure path for transporting packets through an otherwise public network by encapsulating packets inside a transport protocol known as an *IP encapsulation protocol*. Generic routing encapsulation (GRE) is an IP encapsulation protocol that is used to transport packets over a network. Information is sent from one network to the other through a GRE tunnel.

GRE tunneling is accomplished through routable tunnel endpoints that operate on top of existing physical and other logical endpoints. GRE tunnels connect one endpoint to another and provide a clear data path between them.

This topic describes:

## Configuring a GRE Tunnel Port

To configure GRE tunnels on a router, you convert a network port or uplink port on the router to a GRE tunnel port for tunnel services. Each physical tunnel port, named gr-*fpc*/*pic*/*port*, can have one or more logical interfaces, each of which is a GRE tunnel.

After conversion to a GRE tunnel port, the physical port cannot be used for network traffic.

To configure a GRE tunnel port on an router, you need to create logical tunnel interfaces and the bandwidth in gigabits per second to reserve for tunnel services. Include the `tunnel-services bandwidth (`*1g* / *10g*`)` statement at the `[edit chassis fpc` *slot-number* `pic` *number*`]` hierarchy level.

To configure a GRE tunnel port , use any unused physical port on the router to create a logical tunnel interface as shown below:

```
user@host# edit chassis
fpc 0 {
    pic 0 {
        tunnel-services {
            port port-number;
        }
    }
}
```

This also creates a gr- interface.

## Configuring Tunnels to Use Generic Routing Encapsulation

Normally, a GRE tunnel port comes up as soon as it is configured and stays up as long as a valid tunnel source address exists or an interface is operational. Each logical interface you configure on the port can be configured as the source or as the endpoint of a GRE tunnel.

To configure a tunnel port to use GRE:

1. Configure a physical GRE port with a logical interface name and address:

- For IPv4 over GRE, specify the protocol family `inet`:

```
[edit interfaces]
user@host# set gr-fpc/pic/port unit number family inet address
```

- For IPv6 over GRE, specify the protocol family `inet6`:

```
[edit interfaces]
user@host# set gr-fpc/pic/port unit number family inet6 address
```

2. Specify the tunnel source address for the logical interface:

```
[edit interfaces]
user@host# set gr-fpc/pic/port unit number tunnel source source-address
```

3. Specify the destination address:

```
[edit interfaces]
user@host# set gr-fpc/pic/port unit number tunnel destination destination-address
```

RELATED DOCUMENTATION

Configuring Unicast Tunnels | 74

## GRE Keepalive Time Overview

Generic routing encapsulation (GRE) tunnel interfaces do not have a built-in mechanism for detecting when a tunnel is down. You can enable keepalive messages to serve as the detection mechanism.

When you enable a GRE tunnel interface for keepalive messages, the interface sends out keepalive request packets to the remote endpoint at regular intervals. If the data path forwarding for the GRE tunnel works correctly at all points, keepalive response packets are returned to the originator. These keepalive messages are processed by the Routing Engine.

You can configure keepalive messages on the physical or logical GRE tunnel interface. If configured on the physical interface, keepalive messages are sent on all logical interfaces that are part of the physical interface. If configured on an individual logical interface, keepalives are sent only on that logical interface.

You configure how often keepalive messages are sent and the length of time that the interface waits for a keepalive response before marking the tunnel as operationally down.

The keepalive request packet is shown in Figure 3 on page 65.

**Figure 3: Keepalive Request Packet**


The keepalive payload includes information to ensure the keepalive response is correctly delivered to the application responsible for the GRE keepalive process.

The outer GRE header includes:

- Source IP Address—IP address of the endpoint that initiates the keepalive request
- Destination IP Address—IP address of the endpoint that receives the keepalive request
- GRE Protocol ID—IP

The inner GRE header includes:

- Source IP Address—IP address of the endpoint that receives the keepalive request
- Destination IP Address—IP address of the endpoint that initiates the keepalive request
- GRE Protocol ID—A value that the packet forwarding engine recognizes as a GRE keepalive packet

**NOTE**: Starting in Junos OS Release 17.3R1, you can configure IPv6 generic routing encapsulation (GRE) tunnel interfaces on MX Series routers. This lets you run a GRE tunnel over an IPv6 network. Packet payload families that can be encapsulated within the IPv6 GRE tunnels include IPv4, IPv6, MPLS, and ISO. Fragmentation and reassembly of the IPv6 delivery packets is not supported.

To configure an IPv6 GRE tunnel interface, specify IPv6 addresses for `source` and `destination` at the `[interfaces gr-0/0/0 unit 0 tunnel]` hierarchy level.

Keepalive is not supported for GRE IPv6.

SEE ALSO

keepalive-time | 211
hold-time (OAM) | 205

# Configuring GRE Keepalive Time

IN THIS SECTION



## Configuring Keepalive Time and Hold time for a GRE Tunnel Interface

You can configure the keepalives on a generic routing encapsulation (GRE) tunnel interface by including both the `keepalive-time` statement and the `hold-time` statement at the `[edit protocols oam gre-tunnel interface` *`interface-name`*`]` hierarchy level.

**NOTE**: For proper operation of keepalives on a GRE interface, you must also include the `family inet` statement at the `[edit interfaces` *`interface-name`* `unit` *`unit`*`]` hierarchy level. If you do not include this statement, the interface is marked as down.

To configure a GRE tunnel interface:

1. Configure the GRE tunnel interface at `[edit interfaces` *`interface-name`* `unit` *`unit-number`*`]` hierarchy level, where the interface name is gr-x/y/z, and the family is set as `inet`.

```
user@host# set interfaces interface-name unit unit-number family family-name
```

2. Configure the rest of the GRE tunnel interface options as explained in *Configuring a GRE Tunnel Interface Between a PE and CE Router* or *Configuring a GRE Tunnel Interface Between PE Routers* based on requirement.

To configure keepalive time for a GRE tunnel interface:

1. Configure the Operation, Administration, and Maintenance (OAM) protocol at the `[edit protocols]` hierarchy level for the GRE tunnel interface.

```
[edit]
user@host# edit protocols oam
```

2. Configure the GRE tunnel interface option for OAM protocol.

```
[edit protocols oam]
user@host# edit gre-tunnel interface interface-name
```

3. Configure the keepalive time from 1 through 50 seconds for the GRE tunnel interface.

```
[edit protocols oam gre-tunnel interface interface-name]
user@host# set keepalive-time seconds
```

4. Configure the hold time from 5 through 250 seconds. Note that the hold time must be at least twice the keepalive time.

```
[edit protocols oam gre-tunnel interface interface-name]
user@host# set hold-time seconds
```

## Display GRE Keepalive Time Configuration

**IN THIS SECTION**



### Purpose

Display the configured keepalive time value as 10 and hold time value as 30 on a GRE tunnel interface (for example, gr-1/1/10.1).

#### Action

To display the configured values on the GRE tunnel interface, run the `show oam gre-tunnel` command at the `[edit protocols]` hierarchy level:

```
[edit protocols]
user@host# show oam gre-tunnel
    interface gr-1/1/10.1 {
        keepalive-time 10;
        hold-time 30;
    }
```

### Display Keepalive Time Information on a GRE Tunnel Interface

IN THIS SECTION



#### Purpose

Display the current status information of a GRE tunnel interface when keepalive time and hold time parameters are configured on it and when the hold time expires.

#### Action

To verify the current status information on a GRE tunnel interface (for example, gr-3/3/0.3), run the `show interfaces gr-3/3/0.3 terse` and `show interfaces gr-3/3/0.3 extensive` operational commands.

**show interfaces gr-3/3/0.3 terse**

```
user@host> show interfaces gr-3/3/0.3 terse

Interface               Admin Link Proto    Local                 Remote
gr-3/3/0.3              up    up   inet     192.0.2.1/24
                                   mpls
```

### show interfaces gr-3/3/0.3 extensive

```
user@host> show interfaces gr-3/3/0.3 extensive
Logical interface gr-3/3/0.3 (Index 73) (SNMP ifIndex 594) (Generation 900)
    Flags: Point-To-Point SNMP-Traps 0x4000 IP-Header
10.1.19.11:10.1.19.12:47:df:64:0000000000000000 Encapsulation: GRE-NULL
    Gre keepalives configured: On, Gre keepalives adjacency state: down
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
    Traffic statistics:
     Input  bytes  :            15629992
     Output bytes  :            15912273
     Input  packets:              243813
     Output packets:              179476
    Local statistics:
     Input  bytes  :            15322586
     Output bytes  :            15621359
     Input  packets:              238890
     Output packets:              174767
    Transit statistics:
     Input  bytes  :              307406                    0 bps
     Output bytes  :              290914                    0 bps
     Input  packets:                4923                    0 pps
     Output packets:                4709                    0 pps
    Protocol inet, MTU: 1476, Generation: 1564, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
        Destination: 192.0.2/24, Local: 192.0.2.1, Broadcast: 192.0.2.255, Generation: 1366
    Protocol mpls, MTU: 1464, Maximum labels: 3, Generation: 1565, Route table: 0
```

**NOTE**: When the hold time expires:

- The GRE tunnel will stay up even though the interface cannot send or receive traffic.
- The `Link` status will be `Up` and the `Gre keepalives adjacency state` will be `Down`.

#### Meaning

The current status information of a GRE tunnel interface with keepalive time and hold time parameters is displayed as expected when the hold time expires.

RELATED DOCUMENTATION

keepalive-time | 211
hold-time (OAM) | 205

## Enabling Fragmentation on GRE Tunnels

To enable fragmentation of IPv4 packets in generic routing encapsulation (GRE) tunnels, include the `clear-dont-fragment-bit` statement and a maximum transmission unit (MTU) setting for the tunnel as part of an existing GRE configuration at the `[edit interfaces]` hierarchy level:

```
[edit interfaces]
gr-fpc/pic/port {
    unit logical-unit-number {
        clear-dont-fragment-bit;
        ...
        family inet {
            mtu 1000;
            ...
        }
    }
}
```

This statement clears the Don't Fragment (DF) bit in the packet header, regardless of the packet size. If the packet size exceeds the tunnel MTU value, the packet is fragmented before encapsulation. The maximum MTU size configurable on the AS or Multiservices PIC is 9192 bytes.

**NOTE**: The `clear-dont-fragment-bit` statement is supported only on MX Series routers and all M Series routers except the M320 router.

**NOTE**: On SRX platforms the clearing of the DF bit on a GRE tunnel is supported only when the device is in packet or selective packet mode; This feature is not supported in flow mode. As a result, when in flow mode, a packet that exceeds the MTU of the GRE interface with the DF bit set is dropped, despite having the `clear-dont-fragment-bit` configured on the GRE interface.

Fragmentation is enabled only on IPv4 packets being encapsulated in IPv4-based GRE tunnels.

**NOTE**: This configuration is supported only on GRE tunnels on AS or Multiservices interfaces. If you commit `gre-fragmentation` as the encapsulation type on a standard Tunnel PIC interface, the following console log message appears when the PIC comes online:

```
gr-fpc/pic/port: does not support this encapsulation
```

The Packet Forwarding Engine updates the IP identification field in the outer IP header of GRE-encapsulated packets, so that reassembly of the packets is possible after fragmentation. The previous CLI constraint check that required you to configure either the `clear-dont-fragment-bit` statement or a tunnel key with the `allow-fragmentation` statement is no longer enforced.

When you configure the `clear-dont-fragment-bit` statement on an interface with the MPLS protocol family enabled, you must specify an MTU value. This MTU value must not be greater than maximum supported value (9192).

SEE ALSO

Configuring Unicast Tunnels | 74

# Configuring IP Tunnel Interfaces

IN THIS SECTION



## Configuring IPv6-over-IPv4 Tunnels

If you have a Tunnel PIC installed in your M Series or T Series router, you can configure IPv6-over-IPv4 tunnels. To define a tunnel, you configure a unicast tunnel across an existing IPv4 network infrastructure. IPv6/IPv4 packets are encapsulated in IPv4 headers and sent across the IPv4 infrastructure through the configured tunnel. You manually configure configured tunnels on each end point.

On SRX Series Firewalls, Generic Routing Encapsulation (GRE) and IP-IP tunnels use internal interfaces, gr-0/0/0 and ip-0/0/0, respectively. The Junos OS creates these interfaces at system bootup; they are not associated with a physical interface.

IPv6-over-IPv4 tunnels are defined in RFC 2893, *Transition Mechanisms for IPv6 Hosts and Routers*. For information about configuring a unicast tunnel, see "Configuring Unicast Tunnels" on page 74. For an IPv6-over-IPv4 tunnel configuration example, see "Tunnel Services Overview" on page 2.

#### SEE ALSO

Tunnel Services Overview | 2

## Example: Configuring an IPv6-over-IPv4 Tunnel

Configure a tunnel on both sides of the connection.

**Configuration on Router 1**

```
[edit]
interfaces {
    gr-1/0/0 {
        unit 0 {
            tunnel {
                source 10.19.2.1;
                destination 10.19.3.1;
            }
            family inet6 {
                address 2001:DB8::1:1/126;
            }
        }
    }
}
```

Configuration on Router 2

```
[edit]
interfaces {
    gr-1/0/0 {
        unit 0 {
            tunnel {
                source 10.19.3.1;
                destination 10.19.2.1;
            }
            family inet6 {
                address 2001:DB8::2:1/126;
            }
        }
    }
}
```

SEE ALSO

Tunnel Services Overview | 2

# Filtering Unicast Packets Through Multicast Tunnel Interfaces

IN THIS SECTION

# Configuring Unicast Tunnels

IN THIS SECTION



To configure a unicast tunnel, you configure a `gr-` interface (to use GRE encapsulation) or an `ip-` interface (to use IP-IP encapsulation) and include the `tunnel` and `family` statements:

```
gr-fpc/pic/port or ip-fpc/pic/port {
    unit logical-unit-number {
        copy-tos-to-outer-ip-header;
        reassemble-packets;
        tunnel {
            allow-fragmentation;
            destination destination-address;
            do-not-fragment;
            key number;
            routing-instance {
                destination routing-instance-name;
            }
            source address;
            ttl number;
        }
        family family {
            address address {
                destination address;
            }
        }
    }
}
```

You can configure these statements at the following hierarchy levels:

- `[edit interfaces]`
- `[edit logical-systems *logical-system-name* interfaces]`

You can configure multiple logical units for each GRE or IP-IP interface, and you can configure only one tunnel per unit.

> **NOTE**: On M Series and T Series routers, you can configure the interface on a service PIC or a tunnel PIC. On MX Series routers, configure the interface on a Multiservices DPC.

Each tunnel interface must be a point-to-point interface. Point to point is the default interface connection type, so you do not need to include the `point-to-point` statement in the logical interface configuration.

You must specify the tunnel's destination and source addresses. The remaining statements are optional.

> **NOTE**: For transit packets exiting the tunnel, forwarding path features, such as reverse path forwarding (RPF), forwarding table filtering, source class usage, destination class usage, and stateless firewall filtering, are not supported on the interfaces you configure as tunnel sources, but are supported on tunnel-pic interfaces.
> However, class-of-service (CoS) information obtained from the GRE or IP-IP header is carried over the tunnel and is used by the re-entering packets. For more information, see the Junos OS Class of Service User Guide for Routing Devices.
>
> To prevent an invalid configuration, the Junos OS disallows setting the address specified by the `source` or `destination` statement at the `[edit interfaces gr-*fpc*/*pic*/*port* unit *logical-unit-number* tunnel]` hierarchy level to be the same as the interface's own subnet address, specified by the `address` statement at the `[edit interfaces gr-*fpc*/*pic*/*port* unit *logical-unit-number* family *family-name*]` hierarchy level.

To set the time-to-live (TTL) field that is included in the encapsulating header, include the `ttl` statement. If you explicitly configure a TTL value for the tunnel, you must configure it to be one larger than the number of hops in the tunnel. For example, if the tunnel has seven hops, you must configure a TTL value of 8.

You must configure at least one family on the logical interface. To enable MPLS over GRE tunnel interfaces, you must include the `family mpls` statement in the GRE interface configuration. In addition, you must include the appropriate statements at the `[edit protocols]` hierarchy level to enable Resource Reservation Protocol (RSVP), MPLS, and label-switched paths (LSPs) over GRE tunnels. Unicast tunnels are bidirectional.

A configured tunnel cannot go through Network Address Translation (NAT) at any point along the way to the destination. For more information, see "Tunnel Services Overview" on page 2 and the MPLS Applications User Guide.

For a GRE tunnel, the default is to set the ToS bits in the outer IP header to all zeros. To have the Routing Engine copy the ToS bits from the inner IP header to the outer, include the `copy-tos-bits-to-outer-ip-header` statement. (This inner-to-outer ToS bits copying is already the default behavior for IP-IP tunnels.)

For GRE tunnel interfaces on Adaptive Services or Multiservices interfaces, you can configure additional tunnel attributes, as described in the following sections:

## Configuring a Key Number on GRE Tunnels

For Adaptive Services and Multiservices interfaces on M Series and T Series routers, you can assign a key value to identify an individual traffic flow within a GRE tunnel, as defined in RFC 2890, *Key and Sequence Number Extensions to GRE*. However, only one key is allowed for each tunnel source and destination pair.

Each IP version 4 (IPv4) packet entering the tunnel is encapsulated with the GRE tunnel key value. Each IPv4 packet exiting the tunnel is verified by the GRE tunnel key value and de-encapsulated. The Adaptive Services or Multiservices PIC drops packets that do not match the configured key value.

To assign a key value to a GRE tunnel interface, include the `key` statement:

```
key number;
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces interface-name unit logical-unit-number tunnel]`
- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number tunnel]`

The key number can be 0 through 4,294,967,295. You must configure the same GRE tunnel key value on tunnel endpoints.

The following example illustrates the use of the key statement in a GRE tunnel configuration:

```
interfaces {
    gr-1/2/0 {
        unit 0 {
            tunnel {
                source 10.58.255.193;
                destination 10.58.255.195;
```

```
                key 1234;
            }
            ...
            family inet {
                mtu 1500;
                address 10.200.0.1/30;
                ...
            }
        }
    }
}
```

## Enabling Packet Fragmentation on GRE Tunnels Prior to GRE Encapsulation

For GRE tunnel interfaces on Adaptive Services and Multiservices interfaces only, you can enable fragmentation of IPv4 packets before they are GRE-encapsulated in GRE tunnels.

By default, IPv4 traffic transmitted over GRE tunnels is not fragmented. To enable fragmentation of IPv4 packets in GRE tunnels, include the `clear-dont-fragment-bit` statement:

```
clear-dont-fragment-bit;
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]`

When you include the `clear-dont-fragment-bit` statement in the configuration, the don't-fragment (DF) bit is cleared on all packets, even packets that do not exceed the tunnel maximum transmission unit (MTU). If the packet's size exceeds the tunnel's MTU value, the packet is fragmented before encapsulation. If the packet's size does not exceed the tunnel's MTU value, the packet is not fragmented.

You can also clear the DF bit in packets transmitted over IP Security (IPsec) tunnels. For more information, see *Configuring IPsec Rules*.

## Specifying an MTU Setting for the Tunnel

To enable key numbers and fragmentation on GRE tunnels (as described in "Configuring a Key Number on GRE Tunnels" on page 76 and "Enabling Packet Fragmentation on GRE Tunnels Prior to GRE Encapsulation" on page 77), you must also specify an MTU setting for the tunnel.

To specify an MTU setting for the tunnel, include the `mtu` statement:

```
mtu bytes;
```

You can include this statement at the following hierarchy levels:

- [edit interfaces gr-*fpc*/*pic*/*port* unit *logical-unit-number* family inet]
- [edit logical-system *logical-system-name* interfaces gr-*fpc*/*pic*/*port* unit *logical-unit-number* family inet]

For more information about MTU settings, see the Junos OS Network Interfaces Library for Routing Devices.

## Configuring a GRE Tunnel to Copy ToS Bits to the Outer IP Header

Unlike IP-IP tunnels, GRE tunnels do not copy the ToS bits to the outer IP header by default. To have the Routing Engine copy the inner ToS bits to the outer IP header (which is required for some tunneled routing protocols) on packets sent by the Routing Engine, include the `copy-tos-to-outer-ip-header` statement at the logical unit hierarchy level of a GRE interface. This example copies the inner ToS bits to the outer IP header on a GRE tunnel:

```
[edit interfaces]
gr-0/0/0 {
    unit 0 {
        copy-tos-to-outer-ip-header;
        family inet;
    }
}
```

## Enabling Fragmentation and Reassembly on Packets After GRE-Encapsulation

You can enable the fragmentation and reassembly of packets after they are GRE-encapsulated for a GRE tunnel. When the size of a GRE-encapsulated packet is greater than the MTU of a link that the packet passes through, the GRE-encapsulated packet is fragmented. You configure the GRE interface at the endpoint of the tunnel to reassemble the fragmented GRE-encapsulated packets before they are processed further on the network.

For each tunnel you configure on an interface, you can enable or disable fragmentation of GRE-encapsulated packets by including the `allow-fragmentation` or `do-not-fragment` statement:

```
allow-fragmentation;
do-not-fragment;
```

You can configure these statements at the following hierarchy levels:

- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `tunnel]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `tunnel]`

If you configure `allow-fragmentation` on a tunnel, the DF bit is not set in the outer IP header of the GRE-encapsulated packet, enabling fragmentation. By default, GRE-encapsulated packets that exceed the MTU size of a link are not fragmented and are dropped.

To enable reassembly of fragmented GRE-encapsulated packets on the GRE interface at the endpoint of the tunnel, include the `reassemble-packets` statement:

```
reassemble-packets;
```

You can configure this statement at the following hierarchy levels:

- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]`

Starting with Junos OS Release 17.3R1, you can configure fragmentation and reasssembly of GRE-encapsulated packets on GRE tunnel interfaces on MX Series routers with MPC7Es, MPC8Es, and MPC9Es.

Starting with Junos OS Release 17.1R1, you can configure fragmentation and reasssembly of GRE-encapsulated packets on GRE tunnel interfaces on MX Series routers with MPC2E-NGs, MPC3E-NGs, MPC5Es, and MPC6Es.

Starting with Junos OS Release 14.2, you can configure fragmentation and reasssembly of GRE-encapsulated packets on GRE tunnel interfaces on MX Series routers with MPC1s, MPC2s, MPC3s, MPC4s, and MPC-16X10GEs.

In Junos OS Release Release 14.1 and earlier, fragmentation and reasssembly of GRE-encapsulated packets is supported only on MX Series routers with MS-DPCs.

### Support for IPv6 GRE tunnels

Starting in Junos OS Release 17.3R1, you can configure IPv6 generic routing encapsulation (GRE) tunnel interfaces on MX Series routers. This lets you run a GRE tunnel over an IPv6 network. Packet payload families that can be encapsulated within the IPv6 GRE tunnels include IPv4, IPv6, MPLS, and ISO. Fragmentation and reassembly of the IPv6 delivery packets is not supported.

To configure an IPv6 GRE tunnel interface, specify IPv6 addresses for `source` and `destination` at the `[interfaces gr-0/0/0 unit 0 tunnel]` hierarchy level, specify `family inet6` at the `[interfaces gr-0/0/0 unit 0]` hierarchy level, and specify an IPv6 address for `address` at the `[interfaces gr-0/0/0 unit 0 family inet6]` hierarchy level.

#### SEE ALSO

Tunnel Services Overview | 2

## Examples: Configuring Unicast Tunnels

Configure two unnumbered IP-IP tunnels:

```
[edit interfaces]
ip-0/3/0 {
    unit 0 {
        tunnel {
            source 192.168.4.18;
            destination 192.168.4.253;
        }
        family inet;
    }
    unit 1 {
        tunnel {
            source 192.168.4.18;
            destination 192.168.4.254;
        }
        family inet;
    }
}
```

Configure numbered tunnel interfaces by including an address at the `[edit interfaces ip-0/3/0 unit (0 | 1) family inet]` hierarchy level:

```
[edit interfaces]
ip-0/3/0 {
    unit 0 {
        tunnel {
            source 192.168.4.18;
            destination 192.168.4.253;
        }
        family inet {
            address 10.5.5.1/30;
        }
    }
    unit 1 {
        tunnel {
            source 192.168.4.18;
            destination 192.168.4.254;
        }
        family inet {
            address 10.6.6.100/30;
        }
    }
}
```

Configure an MPLS over GRE tunnel by including the `family mpls` statement at the `[edit interfaces gr-1/2/0 unit 0]` hierarchy level:

```
[edit interfaces]
gr-1/2/0 {
    unit 0 {
        tunnel {
            source 192.168.1.1;
            destination 192.168.1.2;
        }
        family inet {
            address 10.1.1.1/30;
        }
        family mpls;
    }
}
```

SEE ALSO

Tunnel Services Overview | 2

## Restricting Tunnels to Multicast Traffic

For interfaces that carry IPv4 or IP version 6 (IPv6) traffic, you can configure a tunnel interface to allow multicast traffic only. To configure a multicast-only tunnel, include the `multicast-only` statement:

```
multicast-only;
```

You can configure this statement at the following hierarchy levels:

- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `family` *`family`*`]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `family` *`family`*`]`

Multicast tunnels filter all unicast packets; if an incoming packet is not destined for a 224/8 or greater prefix, the packet is dropped and a counter is incremented.

You can configure this property on GRE, IP-IP, PIM, and multicast tunnel (`mt`) interfaces only.

> **NOTE**: If your router has a Tunnel Services PIC, the Junos OS automatically configures one multicast tunnel interface (`mt`) for each virtual private network (VPN) you configure. You do not need to configure multicast tunnel interfaces.

SEE ALSO

Tunnel Services Overview | 2

# Connecting Logical Systems Using Logical Tunnel Interfaces

IN THIS SECTION



## Configuring Logical Tunnel Interfaces

IN THIS SECTION



Logical tunnel (`lt-`) interfaces provide quite different services depending on the host router:

- On M Series, MX Series, and T Series routers, logical tunnel interfaces allow you to connect logical systems, virtual routers, or VPN instances. M Series and T Series routers must be equipped with a Tunnel Services PIC or an Adaptive Services Module (only available on M7i routers). MX Series routers must be equipped with a Trio MPC/MIC module. For more information about connecting these applications, see the Junos OS VPNs Library for Routing Devices.

- On SRX Series Firewalls, the logical tunnel interface is used to interconnect logical systems. See the Logical Systems and Tenant Systems User Guide for Security Devices for information about using the logical tunnel interface on the SRX Series.

- On ACX Series routers, logical tunnel interfaces allow you to connect a bridge domain and a pseudowire. Logical systems are not supported on ACX Series routers.

## Connecting Logical Systems

To connect two logical systems, you configure a logical tunnel interface on both logical systems. Then you configure a peer relationship between the logical tunnel interfaces, thus creating a point-to-point connection.

To configure a point-to-point connection between two logical systems, configure the logical tunnel interface by including the `lt-`*fpc/pic/port* statement:

```
lt-fpc/pic/port {
    unit logical-unit-number {
        encapsulation encapsulation;
        peer-unit unit-number;   # peering logical system unit number
        dlci dlci-number;
        family (inet | inet6 | iso | mpls);
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces]`
- `[edit logical-systems` *logical-system-name* `interfaces]`

When configuring logical tunnel interfaces, note the following:

- You can configure each logical tunnel interface with one of the following encapsulation types: Ethernet, Ethernet circuit cross-connect (CCC), Ethernet VPLS, Frame Relay, Frame Relay CCC, VLAN, VLAN CCC, or VLAN VPLS.
- You can configure the IP, IPv6, International Organization for Standardization (ISO), or MPLS protocol family.
- Do not reconfigure a logical tunnel interface that is an anchor point with pseudowire devices stacked above it unless you first deactivate all broadband subscribers that are using the pseudowire subscriber interface.
- The peering logical interfaces must belong to the same logical tunnel interface derived from the Tunnel Services PIC or Adaptive Services Module.
- You can configure only one peer unit for each logical interface. For example, unit 0 cannot peer with both unit 1 and unit 2.

- To enable the logical tunnel interface, you must configure at least one physical interface statement.
- Logical tunnels are not supported with Adaptive Services, Multiservices, or Link Services PICs (but they are supported on the Adaptive Services Module on M7i routers, as noted above).
- On M Series routers other than the M40e router, logical tunnel interfaces require an Enhanced Flexible PIC Concentrator (FPC).
- On MX Series routers, logical tunnel interfaces require Trio MPC/MIC modules. They do not require a Tunnel Services PIC in the same system.

SEE ALSO

Tunnel Services Overview | 2

## Guidelines for Configuring Logical Tunnels on MX Series Routers

When you configure a logical tunnel on an MX series router which has one of the peer configured in layer 2 mode, ensure that the peer layer 2 logical tunnel is part of a bridge domain or VPLS instance, for bidirectional traffic flow.

To configure a logical tunnel with bridge encapsulation, you must first configure the logical tunnel to be part of the bridge domain. The following sample configuration allows you to configure a logical tunnel, lt-2/1/0.3 with bridge encapsulation.

```
user@host# edit bridge-domains {
             bd1 {
                    domain-type bridge;
                    vlan-id 1
                   }
                            }


user@host# edit chassis
lt-2/1/0 {
    unit 3 {
        description "MPLS port mirroring Bridge ingress interface";
        encapsulation ethernet-bridge;
                   mtu 4500;
```

```
            peer-unit 4;
            family bridge {
                interface-mode access;
                vlan-id 1;
                          }
            }

    unit 4 {
            description "MPLS Port mirroring L2/CCC egress interface";
            encapsulation ethernet-ccc;
            mtu 4500;
            peer-unit 3;
            family ccc {
                filter {
                    input HighPriority;
                }
            }
          }
    }
```

## Guidelines for Configuring Logical Tunnels on ACX Series Routers

Observe the following guidelines while configuring logical tunnel (`lt-`) interfaces on ACX Series routers:

- You can use a logical tunnel interface to connect only bridge domains and pseudowires.
- Logical tunnel interfaces cannot interconnect the following links:
  - Pesudowire and a routing instance (Pseudowire terminating on a VRF)
  - Two routing instances
  - VPLS instance and a routing instance
  - Two VPLS instances
  - Two Bridge domains
  - Bridge domain and a VPLS instance
- Only one logical tunnel (physical interface) per bandwidth type (1 Gbps or 10 Gbps) can be configured on ACX routers. However, you can specify up to two logical tunnel interfaces (one with 1 Gb bandwidth and another with 10 Gb bandwidth) on ACX routes.

- Guaranteed bandwidth for logical tunnels is 1 Gbps and certain platforms support up to an additional 10 Gbps bandwidth. All the services configured using logical tunnel interfaces share this bandwidth.

  The bandwidth configured on the logical tunnel interface is shared between upstream and downstream traffic on that interface. The effective bandwidth available for the service is half the configured bandwidth.

- Multiple logical tunnel interfaces to enable configuration of separate services on each logical interface to obtain increased bandwidth for each individual interface separately or the bundling of individual logical tunnel interfaces is not supported.

- You can configure Ethernet VLAN, Ethernet CCC, VLAN bridge on Ethernet interfaces, and VLAN on circuit cross-connects (CCC) as encapsulation types on logical tunnel interfaces. Other encapsulation types such as Ethernet, VLAN, Ethernet VPLS, or VLAN VPLS are not supported.

- When the encapsulation configured on the logical interface units is one of the supported types such as Ethernet VLAN or VLAN bridge, you can enable only bridge domains or CCC protocols on logical tunnel interfaces. Other address families or protocols such as IPv4, IPv5, MPLS, or OSPF are not supported.

- Classifier, rewrite and ingress policer configuration are supported on logical tunnel interfaces. Fixed, BA-based, and multifield classifiers are supported on the lt- interfaces at the physical interface-level.

  802.1p, 802.1ad, TOS and DSCP based BA classifiers are supported. Remarking rules can be configured at the port level on the LT interface. 802.1p, 802.1ad, TOS and DSCP fields in the packet can be rewritten in the LT interface. Ingress policers are supported.

  Simple, Single-rate tricolor marking (srTCM), two-rate tricolor marking (trTCM) policers are supported. Egress policers are not supported.

- Default classifiers do not work properly when lt- interfaces are configured on non-Ethernet PICs.

- Port-level queuing is supported; up to eight queues per lt- interface are supported. These eight queues are shared between the upstream and downstream traffic traversing through the lt- interface. If the configured bandwidth on the lt- interface is not adequate for the upstream and downstream traffic of the services configured on the interface, a failure occurs with traffic propagation because multiple lt- interfaces are not supported.

- Eight forwarding classes (0-7) are mapped to the eight queues based on the global system configuration. The remainder of the scheduler configuration, buffer-size, transmit-rate, shaping-rate, priority and WRED or drop profiles maps can be configured on the lt- interface queues.

- The following firewall filter types are supported on lt- interfaces:

  - Logical interface-level filters

  - Bridge family filters

- CCC family filters

  All firewall configurations are supported. The scaling limitation with such filters is the same as the existing firewall filter restrictions.

- OAM is not supported on lt- interfaces.
- Similar to other physical interfaces, the number of logical interfaces that can be supported on logical tunnel physical interfaces is 30.
- When a bridge domain is configured with a VLAN ID (bridge domain has normalized VLANs), the difference is behavior between MX and ACX Series routers is that the MX router does not match the user-vlan-id in output filter, whereas the ACX router matches the user-vlan-id specified in the output filter.
- If the logical tunnel interface is created using non Ethernet PICs, then default classifier is not bound to the interface.

To create logical tunnel interfaces and the bandwidth in gigabits per second to reserve for tunnel services, include the `tunnel-services bandwidth (1g | 10g)` statement at the `[edit chassis fpc slot-number pic number]` hierarchy level:

```
[edit interfaces]
    lt-fpc/pic/port {
        unit logical-unit-number {
            encapsulation encapsulation;
            peer-unit unit-number;    # peering logical system unit number
            dlci dlci-number;
            family (inet | inet6 | iso | mpls);
        }
    }
```

The ACX5048 and ACX5096 routers support `ethernet-vpls` and `vlan-vpls` encapsulations. These encapsulations are supported only on logical tunnel interface and are required for configuring hierarchial VPLS.

You can use any unused physical port on the ACX5048 and ACX5096 routers to create a logical tunnel interface as shown below:

```
user@host# edit chassis
fpc 0 {
    pic 0 {
        tunnel-services {
            port port-number;
```

```
        }
    }
}
```

The following sample configuration allows you to encapsulate `vlan-ccc` to `vlan-vpls` using LT interface in ACX5048 and ACX5096 routers:

```
user@host# edit interfaces
lt-0/0/1 {
    unit 0 {
        encapsulation vlan-ccc;
        vlan-id 1;
        peer-unit 1;
    }
    unit 1 {
        encapsulation vlan-vpls;
        vlan-id 1;
        peer-unit 0;
    }
}
```

## Example: Configuring Logical Tunnels

Configure three logical tunnels:

```
[edit interfaces]
lt-4/2/0 {
    description “Logical tunnel interface connects three logical systems”;
}
[edit logical-systems]
lr1 {
    interfaces lt-4/2/0 {
        unit 12 {
            peer-unit 21; #Peering with lr2
            encapsulation frame-relay;
            dlci 612;
            family inet;
        }
```

```
        unit 13 {
            peer-unit 31; #Peering with lr3
            encapsulation frame-relay-ccc;
            dlci 613;
        }
    }
}
lr2 {
    interfaces lt-4/2/0 {
        unit 21 {
            peer-unit 12; #Peering with lr1
            encapsulation frame-relay-ccc;
            dlci 612;
        }
        unit 23 {
            peer-unit 32; #Peering with lr3
            encapsulation frame-relay;
            dlci 623;
        }
    }
}
lr3 {
    interfaces lt-4/2/0 {
        unit 31 {
            peer-unit 13; #Peering with lr1
            encapsulation frame-relay;
            dlci 613;
            family inet;
        }
        unit 32 {
            peer-unit 23; #Peering with lr2
            encapsulation frame-relay-ccc;
            dlci 623;
        }
    }
}
```

## SEE ALSO

Tunnel Services Overview | 2

# Configuring an Interface in the VRF Domain to Receive Multicast Traffic

IN THIS SECTION



You can configure an ACX Series router to receive multicast traffic in a VRF domain. In an IPTV solution, IPTV sources and receivers can be spread across different end points of a network in a VRF domain. To receive the multicast traffic at the receiver's side, it is necessary for the multicast traffic to be tunneled across the network to reach the end receiving device or the subscriber. This tunneling is usually done using the Multicast Virtual Private Network (MVPN) technology.

ACX Series routers do not support MVPN technology. An alternate method for receiving the multicast traffic in the VRF domain in ACX Series router is by associating a global logical interface to a logical interface in the VRF domain. The global logical interface acts as a proxy for receiving the multicast traffic on the logical interface in the VRF domain. To associate a global logical interface to a logical interface in the VRF domain, you need to configure an IRB interface in a global domain to act as a proxy for the logical interface in the VRF domain.

## Configuring a Proxy Logical Interface in the Global Domain

To configure a proxy logical interface in the global domain, you need to create logical tunnel (lt-) interface and IRB interface and then associate the IRB interface to a bridge domain. The following is a example to configure a proxy logical interface in the global domain:

1. Create an logical tunnel (lt-) interface.

```
[edit]
user@host# set chassis aggregated-devices ethernet device-count 1
user@host# set chassis fpc 0 pic 0 tunnel-services bandwidth 1g
user@host# set interfaces lt-0/0/10 unit 0 encapsulation vlan-bridge
user@host# set interfaces lt-0/0/10 unit 0 vlan-id 101
user@host# set interfaces lt-0/0/10 unit 0 peer-unit 1
```

```
user@host# set interfaces lt-0/0/10 unit 1 encapsulation vlan-ccc
user@host# set interfaces lt-0/0/10 unit 1 vlan-id 101
user@host# set interfaces lt-0/0/10 unit 1 peer-unit 0
```

2. Create an IRB interface.

```
[edit]
user@host# set interfaces irb unit 0 family inet address 192.168.1.2/24
```

3. Associate the IRB interface to a bridge domain.

```
[edit]
user@host# set bridge-domains b1 vlan-id 101
user@host# set bridge-domains b1 interface lt-0/0/10.0
user@host# set bridge-domains b1 routing-interface irb.0
```

## Associating the Proxy Logical Interface to a Logical Interface in a VRF Domain

To associate the proxy logical interface to a logical interface in a VRF domain, you need to run the following PFE commands:

- `test pfe acx vrf-mc-leak enable`—Enables proxy association.
- `test pfe acx entry add` *VRF-logical-interface-name logical-tunnel-logical-interface-name IRB-logical-interface-name IRB-IP-address + 1*—Creates an association between proxy logical interface and the logical interface in a VRF domain.
- `test pfe acx vrf-mc-leak disable`—Disables proxy association.
- `test pfe acx entry del` *VRF-logical-interface-name logical-tunnel-logical-interface-name IRB-logical-interface-name IRB-IP-address + 1*—Deletes the association between the proxy logical interface and the logical interface in a VRF domain.
- `show pfe vrf-mc-leak`—Displays the association entries between proxy logical interface and the logical interface in a VRF domain.

**NOTE**: When the router or PFE is rebooted, the proxy associations of logical interfaces is removed and you need to once again create the proxy associations of logical interface.

### Limitations

The following limitations need to be considered for receiving multicast traffic in a VRF domain:

- Maximum of 5 proxy associations of logical interfaces can be configured.
- VRF IPv6 multicast is not supported.
- AE interface as a VRF interface (requesting multicast traffic) is not be supported.
- Multicast traffic cannot be forwarded from the logical interface in a VRF domain if the first hop router is an ACX router.

## Redundant Logical Tunnels Overview

IN THIS SECTION

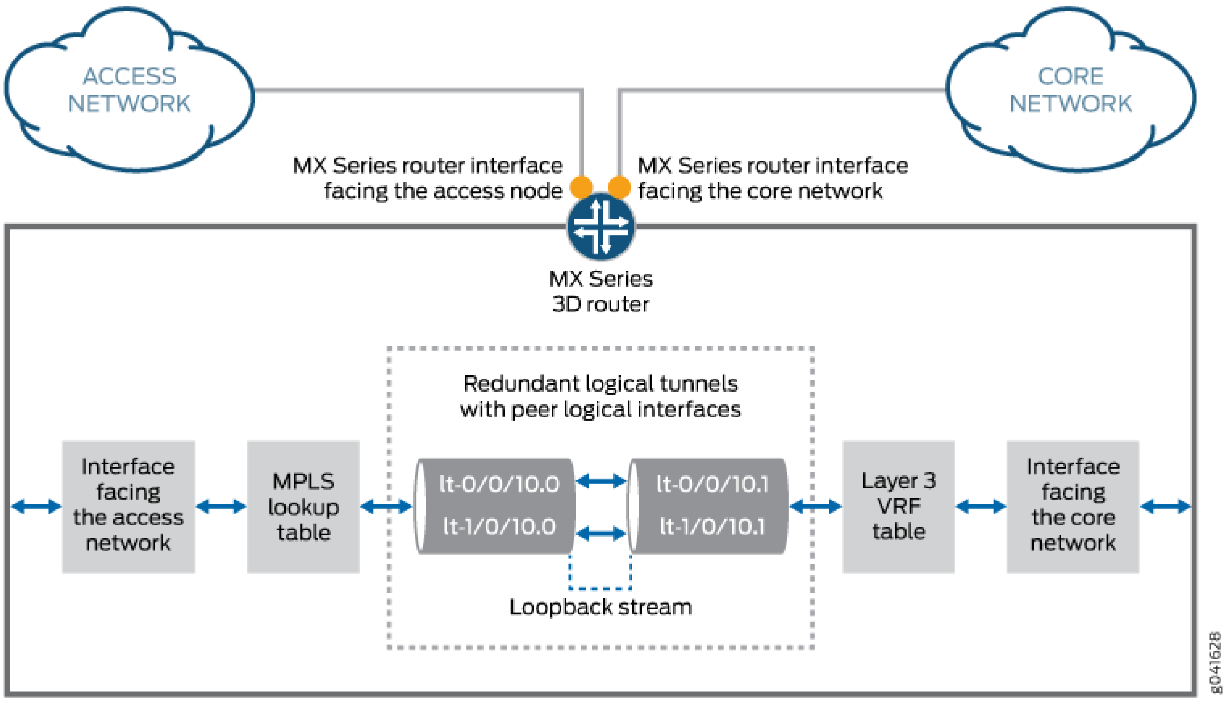


You can connect two devices, such as an access-facing device and a core-facing device, through logical tunnels. To provide redundancy for the tunnels, you can create and configure multiple physical logical tunnels and add them to a virtual redundant logical tunnel.

**NOTE**: Redundant logical tunnels are supported only on MX Series routers with MPCs. Starting in Junos OS Release 18.4R3, redundant logical tunnels are supported on MX Series Virtual Chassis..

For example, in an MPLS access network, you can configure multiple pseudowires between an access node and an MX Series router with MPCs and add them to a redundant logical tunnel. You can then add multiple logical tunnels to the redundant logical tunnel. Figure 4 on page 94 shows a redundant logical tunnel between the access node and the MX Series router.

**Figure 4: Redundant Logical Tunnels**




The redundant logical tunnel has peer logical interfaces at each end, rlt0.0 and rlt0.1. You can configure router features on these interfaces for the redundant logical tunnel and its members.

Each member logical tunnel has peer logical interfaces. In Figure 4 on page 94, lt-0/0/10.0 and lt-0/0/10.1 are peers.

The MX Series router performs IP lookup in the Layer 3 VPN routing and forwarding (VRF) table on the router where the pseudowires that are grouped in logical tunnels terminate.

## Redundant Logical Tunnel Configuration

In Junos OS Releases 14.1R1 and earlier, you can create up to 16 redundant logical tunnels, depending on the number of Packet Forwarding Engines and the number of loopback interfaces on each Packet Forwarding Engine on your device. Starting in Junos OS Release 14.2 and for 13.3R3 and 14.1R2, the valid range for device-count is from 1 to 255.

You can add up to 32 logical tunnels as members of a redundant logical tunnel.

When you add more than two members to the redundant logical tunnel, they are in active mode. The traffic is load-balanced over all the tunnel members.

When you add only two members to the redundant logical tunnel, you can configure the members in one of these ways:

- Both members in active mode
- One member in active mode and the other in backup mode

### Redundant Logical Tunnel Failure Detection and Failover

A logical tunnel fails and is removed from the redundant logical tunnel group, and the backup logical tunnel becomes active due to one of these events:

- A hardware failure on the MPC module occurs.
- An MPC failure occurs due to a microkernel crash.
- The MPC module is administratively shut down and removed from the redundant logical tunnel.
- A power failure on the MPC module occurs.

**NOTE**: You can decrease the time it takes for failure detection and failover to occur. Configure the `enhanced-ip` statement at the `[edit chassis network-services]` hierarchy level to enable Packet Forwarding Engine liveliness detection.

#### SEE ALSO

*Pseudowire Subscriber Logical Interfaces Overview*

*Configuring a Pseudowire Subscriber Logical Interface Device*

## Configuring Redundant Logical Tunnels

Use redundant logical tunnels to provide redundancy for logical tunnels between two devices, such as an access-facing device and a core-facing device.

When configuring redundant logical tunnel interfaces, note the following:

- Starting in Junos OS Release 13.3, you can configure redundant logical tunnels only on MX Series routers with MPCs.

  In Junos OS Releases 14.1R1 and earlier, you can create up to 16 redundant logical tunnels, depending on the number of Packet Forwarding Engines and the number of loopback interfaces on each Packet Forwarding Engine on your device. Starting in Junos OS Release 14.2 and for 13.3R3 and 14.1R2, the valid range for device-count is from 1 to 255. The command is shown below.

`set chassis redundancy-group interface-type redundant-logical-tunnel device-count [number];`

You can add up to 32 logical tunnels as members.

- When a logical tunnel with an existing configuration joins a redundant logical tunnel, you must configure the redundant logical tunnel with the settings from the existing configuration.
- You can add member logical tunnels to a parent logical tunnel for redundancy.
- When you add more than two logical tunnels to the redundant logical tunnel, the members are in active mode by default.
- When you add only two members, you can configure the members in one of these ways:
  - Both members in active mode
  - One member in active mode and the other in backup mode

To configure a redundant logical tunnel between two devices:

1. Create the logical tunnel and redundant logical tunnel interfaces.

```
[edit chassis]
user@host# set redundancy-group interface-type redundant-logical-tunnel device-count count
user@host# set fpc slot-number pic number tunnel-services bandwidth 1g
```

2. Bind the member logical tunnels to the redundant logical tunnel.

```
[edit interfaces]
user@host# set interface-name redundancy-group member-interface interface-name
```

3. Configure the redundant logical tunnel interfaces.
4. Attach the redundant logical tunnel interface to a Layer 2 circuit.
5. Add the peer redundant logical tunnel interface to a Layer 3 VRF instance.
6. Configure MPLS and LDP in the pseudowires and the Layer 3 VPN.

```
[edit protocols]
user@host# set mpls no-cspf
user@host# set mpls interface all
user@host# set ldp interface all
```

7. Configure BGP in the Layer 3 VPN.
8. Configure OSPF on the core-facing interfaces and the router local loopback interface.
9. Set the policy options for BGP.
10. Set the router ID and the autonomous system (AS) number.

## Example: Configuring Redundant Logical Tunnels

**IN THIS SECTION**



This example shows how to configure redundant logical tunnels in an MPLS access network.

### Requirements

In Junos OS Release 13.3 or later, you can configure redundant logical tunnels only on MX Series routers with MPCs.

### Overview

**IN THIS SECTION**



When a logical tunnel with an existing configuration joins a redundant logical tunnel, you must configure the redundant logical tunnel with the settings from the existing configuration.

You can add member logical tunnels to a parent logical tunnel for redundancy.

On MX Series routers with MPCs, you can configure redundant logical tunnels as follows:

- In Junos OS Releases 14.1R1 and earlier, you can create up to 16 redundant logical tunnels, depending on the number of Packet Forwarding Engines and the number of loopback interfaces on each Packet Forwarding Engine on your device. Starting in Junos OS Release 14.2 and for 13.3R3 and 14.1R2, the valid range for device-count is from 1 to 255. The command is shown below.

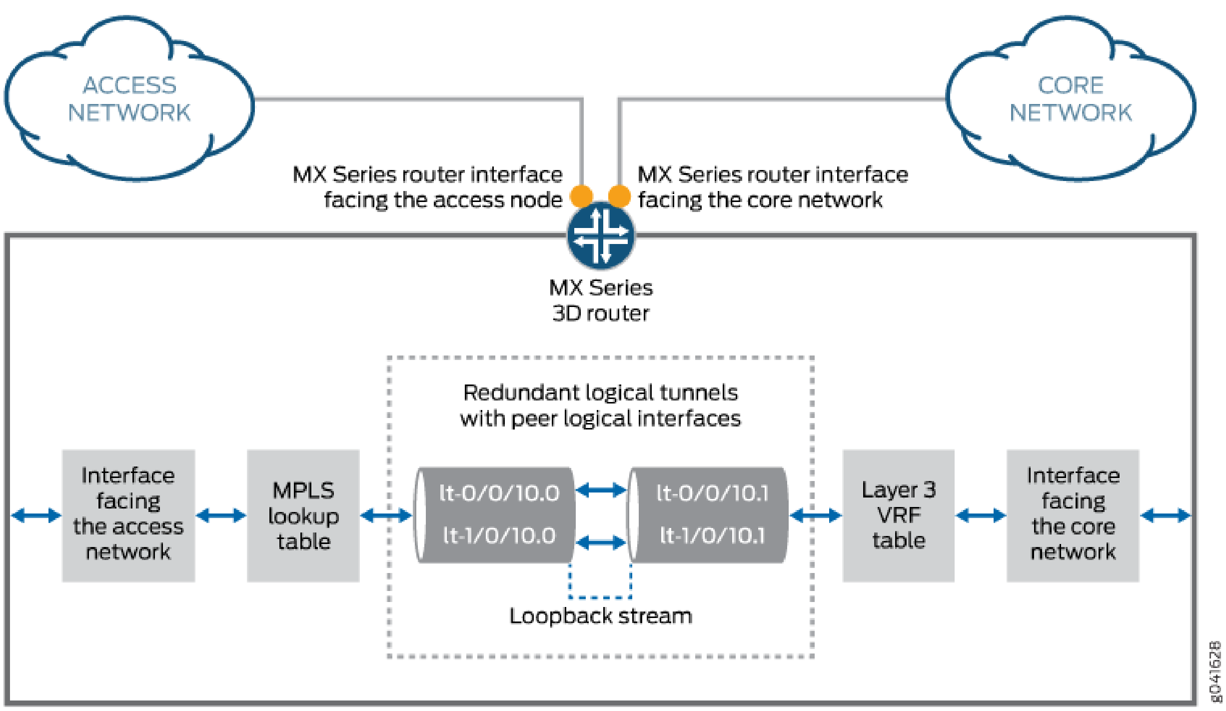
```
set chassis redundancy-group interface-type redundant-logical-tunnel device-count [number];
```

- You can add up to 32 logical tunnels as members.
- When you add more than two logical tunnels to a redundant logical tunnel, the members are in active mode by default.
- When you add only two members, you can configure the members in one of these ways:
  - Both members in active mode
  - One member in active mode and the other in backup mode

**Topology**

Figure 5 on page 98 shows a redundant logical tunnel between the access node and the MX Series router in an MPLS access network.

**Figure 5: Redundant Logical Tunnels**



The redundant logical tunnel has peer logical interfaces at each end, rlt0.0 and rlt0.1. You can configure router features on these interfaces for the redundant logical tunnel and its members.

Each member logical tunnel has peer logical interfaces on the access-facing and core-facing devices. In Figure 5 on page 98, lt-0/0/10.0 and lt-0/0/10.1 are peers.

The MX Series router performs IP lookup in the Layer 3 VPN routing and forwarding (VRF) table on the router where the pseudowires that are grouped in logical tunnels terminate.

## Configuration

**IN THIS SECTION**



### CLI Quick Configuration

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

```
set chassis redundancy-group interface-type redundant-logical-tunnel device-count 4
set chassis fpc 1 pic 0 tunnel-services bandwidth 1g
set chassis fpc 2 pic 2 tunnel-services bandwidth 1g
set interfaces rlt0 redundancy-group member-interface lt-1/0/10
set interfaces rlt0 redundancy-group member-interface lt-2/0/10
set interfaces rlt0 unit 0 description "Towards Layer 2 Circuit"
set interfaces rlt0 unit 0 encapsulation vlan-ccc
set interfaces rlt0 unit 0 vlan-id 600
set interfaces rlt0 unit 0 peer-unit 1
set interfaces rlt0 unit 0 family ccc
set interfaces rlt0 unit 1 description "Towards Layer 3 VRF"
set interfaces rlt0 unit 1 encapsulation vlan
set interfaces rlt0 unit 1 vlan-id 600
set interfaces rlt0 unit 1 peer-unit 0
set interfaces rlt0 unit 1 family inet address 10.10.10.2/24
set protocols l2circuit neighbor 192.0.2.2 interface rlt0.0 virtual-circuit-id 100
```

```
set protocols l2circuit neighbor 192.0.2.2 interface rlt0.0 no-control-word
set routing-instances pe-vrf instance-type vrf
set routing-instances pe-vrf interface rlt0.1
set routing-instances pe-vrf route-distinguisher 65056:1
set routing-instances pe-vrf vrf-import VPN-A-Import
set routing-instances pe-vrf vrf-export VPN-A-Export
set routing-instances pe-vrf vrf-table-label
set routing-instances pe-vrf protocols ospf export VPN-A-Import
set routing-instances pe-vrf protocols ospf area 0.0.0.0 interface rlt0.1
set protocols mpls no-cspf
set protocols mpls interface all
set protocols ldp interface all
set protocols bgp export local-routes
set protocols bgp group internal type internal
set protocols bgp group internal local-address 198.51.100.3
set protocols bgp group internal family inet any
set protocols bgp group internal family inet-vpn unicast
set protocols bgp group internal neighbor 203.0.113.4
set protocols ospf area 0.0.0.0 interface ge-5/3/8.0
set protocols ospf area 0.0.0.0 interface ge-5/2/5.0
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set policy-options policy-statement VPN-A-Export term a then community add VPN-A
set policy-options policy-statement VPN-A-Export term a then accept
set policy-options policy-statement VPN-A-Export term b then reject
set policy-options policy-statement VPN-A-Import term a from protocol bgp
set policy-options policy-statement VPN-A-Import term a from community VPN-A
set policy-options policy-statement VPN-A-Import term a then accept
set policy-options policy-statement VPN-A-Import term b then reject
set policy-options policy-statement local-routes then accept
set policy-options community VPN-A members target:100:100
set routing-options router-id 198.51.100.3
set routing-options autonomous-system 65056
```

#### Procedure

### Step-by-Step Procedure

In this example, all the logical tunnels are in active mode.

1. Create the logical tunnel and redundant logical tunnel interfaces.

```
[edit chassis]
user@host# set redundancy-group interface-type redundant-logical-tunnel device-count 4
user@host# set fpc 1 pic 0 tunnel-services bandwidth 1g
user@host# set fpc 2 pic 2 tunnel-services bandwidth 1g
```

2. Bind the member logical tunnels to the redundant logical tunnel.

```
[edit interfaces]
user@host# set rlt0 redundancy-group member-interface lt-1/0/10
user@host# set rlt0 redundancy-group member-interface lt-2/0/10
```

3. Configure the redundant logical tunnel interfaces.

```
[edit interfaces]
user@host# set rlt0 unit 0 description "Towards Layer 2 Circuit"
user@host# set rlt0 unit 0 encapsulation vlan-ccc
user@host# set rlt0 unit 0 vlan-id 600
user@host# set rlt0 unit 0 peer-unit 1
user@host# set rlt0 unit 0 family ccc
user@host# set rlt0 unit 1 description "Towards Layer 3 VRF"
user@host# set rlt0 unit 1 encapsulation vlan
user@host# set rlt0 unit 1 vlan-id 600
user@host# set rlt0 unit 1 peer-unit 0
user@host# set rlt0 unit 1 family inet address 10.10.10.2/24
```

4. Attach rlt0.0 to a Layer 2 circuit.

```
[edit protocols]
user@host# set l2circuit neighbor 192.0.2.2 interface rlt0.0 virtual-circuit-id 100
user@host# set l2circuit neighbor 192.0.2.2 interface rlt0.0 no-control-word
```

5. Add rlt0.1 to a Layer 3 VRF instance.

```
[edit routing-instances]
user@host# set pe-vrf instance-type vrf
user@host# set pe-vrf interface rlt0.1
```

```
user@host# set pe-vrf route-distinguisher 65056:1
user@host# set pe-vrf vrf-import VPN-A-Import
user@host# set pe-vrf vrf-export VPN-A-Export
user@host# set pe-vrf vrf-table-label
user@host# set pe-vrf protocols ospf export VPN-A-Import
user@host# set pe-vrf protocols ospf area 0.0.0.0 interface rlt0.1
```

6. Configure MPLS and LDP in the pseudowires and the Layer 3 VPN.

```
[edit protocols]
user@host# set mpls no-cspf
user@host# set mpls interface all
user@host# set ldp interface all
```

7. Configure BGP in the Layer 3 VPN.

```
[edit protocols]
user@host# set bgp export local-routes
user@host# set bgp group internal type internal
user@host# set bgp group internal local-address 198.51.100.3
user@host# set bgp group internal family inet any
user@host# set bgp group internal family inet-vpn unicast
user@host# set bgp group internal neighbor 203.0.113.4
```

8. Configure OSPF on the core-facing interfaces and the router local loopback interface.

```
[edit protocols]
user@host# set ospf area 0.0.0.0 interface ge-5/3/8.0
user@host# set ospf area 0.0.0.0 interface ge-5/2/5.0
user@host# set ospf area 0.0.0.0 interface lo0.3 passive
```

9. Set the policy options for BGP.

```
[edit policy-options]
user@host# set policy-statement VPN-A-Export term a then community add VPN-A
user@host# set policy-statement VPN-A-Export term a then accept
user@host# set policy-statement VPN-A-Export term b then reject
user@host# set policy-statement VPN-A-Import term a from protocol bgp
user@host# set policy-statement VPN-A-Import term a from community VPN-A
```

```
user@host# set policy-statement VPN-A-Import term a then accept
user@host# set policy-statement VPN-A-Import term b then reject
user@host# set policy-statement local-routes then accept
user@host# set community VPN-A members target:100:100
```

10. Set the router ID and the autonomous system (AS) number.

```
[edit routing-options]
user@host# set router-id 198.51.100.3
user@host# set autonomous-system 65056
```

### Results

From configuration mode, confirm your configuration by entering the following commands:

- `show chassis`
- `show interfaces`
- `show policy-options`
- `show protocols`
- `show routing-instances`
- `show routing-options`

If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show chassis
redundancy-group {
    interface-type {
        redundant-logical-tunnel {
            device-count 4;
        }
    }
}
fpc 1 {
    pic 0 {
        tunnel-services {
            bandwidth 1g;
        }
```

```
    }
}
fpc 1 {
    pic 2 {
        tunnel-services {
            bandwidth 1g;
        }
    }
}
```

```
user@host# show interfaces rlt0
redundancy-group {
    member-interface lt-1/0/10;
    member-interface lt-2/0/10;
}
unit 0 {
    description "Towards Layer 2 Circuit";
    encapsulation vlan-ccc;
    vlan-id 600;
    peer-unit 1;
    family ccc;
}
unit 1 {
    description "Towards Layer 3 VRF";
    encapsulation vlan;
    vlan-id 600;
    peer-unit 0;
    family inet {
        address 10.10.10.2/24;
    }
}
```

```
user@host# show protocols l2circuit
neighbor 192.0.2.2 {
    interface rlt0.0 {
        virtual-circuit-id 100;
        no-control-word;
```

```
    }
}
```

```
user@host# show protocols
mpls {
    no-cspf;
    interface all;
}
bgp {
    export local-routes;
    group internal {
        type internal;
        local-address 198.51.100.3;
        family inet {
            any;
        }
        family inet-vpn {
            unicast;
        }
        neighbor 203.0.113.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-5/3/8.0;
        interface ge-5/2/5.0;
        interface lo0.3 {
            passive;
        }
    }
}
ldp {
    interface all;
}
l2circuit {
    neighbor 192.0.2.2 {
        interface rlt0.0 {
            virtual-circuit-id 100;
            no-control-word;
        }
```

```
    }
}
```

```
user@host# routing-instances
pe-vrf {
     instance-type vrf;
    interface rlt0.1;
     route-distinguisher 65056:1;
    vrf-import VPN-A-Import;
    vrf-export VPN-A-Export;
    vrf-table-label;
    protocols {
        ospf {
            export VPN-A-Import;
            area 0.0.0.0 {
                interface rlt0.1;
            }
        }
    }
}
```

```
user@host# policy-options
policy-statement VPN-A-Export {
    term a {
        then {
            community add VPN-A;
            accept;
        }
    }
    term b {
        then reject;
    }
}
policy-statement VPN-A-Import {
    term a {
        from {
            protocol bgp;
            community VPN-A;
            }
         then accept;
```

```
    }
    term b {
        then reject;
    }
}
policy-statement local-routes {
    then accept;
}
community VPN-A members target:100:100;
```

```
user@host# routing-options
router-id 198.51.100.3;
autonomous-system 65056;
```

## Verification

IN THIS SECTION



Confirm that the configuration is working properly.

**Verifying the Redundant Logical Tunnel Configuration**

### Purpose

Verify that the redundant logical tunnel with the child logical tunnel interfaces are created with the correct encapsulations.

## Action

```
user@host# run show interfaces terse | match rlt0
lt-1/0/10.0            up    up   container--> rlt0.0
lt-1/0/10.1            up    up   container--> rlt0.1
lt-2/0/10.0            up    up   container--> rlt0.0
lt-2/0/10.1            up    up   container--> rlt0.1
rlt0                   up    up
rlt0.0                 up    up   ccc
rlt0.1                 up    up   inet     10.10.10.2/24
```

### Verifying the Layer 2 Circuit

## Purpose

Verify that the Layer 2 circuit is up.

## Action

```
user@host# run show l2circuit connections
Layer-2 Circuit Connections:

Legend for connection status (St)
EI -- encapsulation invalid      NP -- interface h/w not present
MM -- mtu mismatch               Dn -- down
EM -- encapsulation mismatch     VC-Dn -- Virtual circuit Down
CM -- control-word mismatch      Up -- operational
VM -- vlan id mismatch           CF -- Call admission control failure
OL -- no outgoing label          IB -- TDM incompatible bitrate
NC -- intf encaps not CCC/TCC    TM -- TDM misconfiguration
BK -- Backup Connection          ST -- Standby Connection
CB -- rcvd cell-bundle size bad  SP -- Static Pseudowire
LD -- local site signaled down   RS -- remote site standby
RD -- remote site signaled down  HS -- Hot-standby Connection
XX -- unknown

Legend for interface status
Up -- operational
Dn -- down
Neighbor: 192.0.2.2
```

```
    Interface                 Type  St     Time last up          # Up trans
    rlt0.0(vc 100)            rmt   Up     Aug  8 00:28:04 2013          1
      Remote PE: 192.0.2.2, Negotiated control-word: No
      Incoming label: 299776, Outgoing label: 299776
      Negotiated PW status TLV: No
      Local interface: rlt0.0, Status: Up, Encapsulation: VLAN
```

#### Verifying OSPF Neighbors

### Purpose

Verify that routers are adjacent and able to exchange OSPF data.

### Action

```
user@host# run show ospf neighbor
Address          Interface              State     ID               Pri  Dead
198.168.30.2       ge-5/2/5.0             Full      203.0.113.4        128    38
198.168.20.1       ge-5/3/8.0             Full      192.0.2.2        128    38
```

#### Verifying the BGP Group

### Purpose

Verify that the BGP group is created.

### Action

```
user@host# run show bgp group internal
Group Type: Internal    AS: 65056                  Local AS: 65056
    Name: internal        Index: 0                   Flags: <Export Eval>
    Export: [ local-routes ]
    Holdtime: 0
    Total peers: 1        Established: 1
    203.0.113.4+179
    inet.0: 1/6/3/0
    inet.2: 0/0/0/0
```

```
    bgp.l3vpn.0: 2/2/2/0
    pe-vrf.inet.0: 2/2/2/0
```

### Verifying the BGP Routes in the Routing Table

#### Purpose

Verify that the BGP routes are in the pe-vrf.inet.0 routing table.

#### Action

```
user@host# run show route protocol bgp table pe-vrf.inet.0
pe-vrf.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

198.168.50.0/24     *[BGP/170] 01:18:14, localpref 100, from 203.0.113.4
                      AS path: I, validation-state: unverified
                    > to 198.168.30.2 via ge-5/2/5.0, Push 16
198.168.51.0/24     *[BGP/170] 01:18:14, MED 2, localpref 100, from 203.0.113.4
                      AS path: I, validation-state: unverified
                    > to 198.168.30.2 via ge-5/2/5.0, Push 16
```

**Release History Table**

| Release | Description |
|---|---|
| 18.4R3 | Starting in Junos OS Release 18.4R3, redundant logical tunnels are supported on MX Series Virtual Chassis. |
| 14.2 | Starting in Junos OS Release 14.2 and for 13.3R3 and 14.1R2, the valid range for device-count is from 1 to 255. |
| 14.2 | Starting in Junos OS Release 14.2 and for 13.3R3 and 14.1R2, the valid range for device-count is from 1 to 255. |
| 14.2 | Starting in Junos OS Release 14.2 and for 13.3R3 and 14.1R2, the valid range for device-count is from 1 to 255. The command is shown below. |
| 13.3 | Starting in Junos OS Release 13.3, you can configure redundant logical tunnels only on MX Series routers with MPCs. |

# Configuring Layer 2 Ethernet Services over GRE Tunnel Interfaces

**IN THIS SECTION**



## Layer 2 Services over GRE Tunnel Interfaces on MX Series with MPCs

Starting in Junos OS Release 15.1, you can configure Layer 2 Ethernet services over GRE interfaces (`gr-`*`fpc/pic/port`* to use GRE encapsulation).

Starting in Release 19.1R1, Junos OS supports Layer 2 Ethernet services over GRE interfaces (to use GRE encapsulation) with IPv6 traffic.

The outputs of the `show bridge mac-table` and `show vpls mac-table` commands have been enhanced to display the MAC addresses learned on a GRE logical interface and the status of MAC address learning properties in the MAC address and MAC flags fields. Also, the `L2 Routing Instance` and `L3 Routing Instance` fields are added to the output of the `show interfaces gr` command to display the names of the routing instances associated with the GRE interfaces are displayed.

To enable Layer 2 Ethernet packets to be terminated on GRE tunnels, you must configure the bridge domain protocol family on the `gr-` interfaces and associate the `gr-` interfaces with the bridge domain. You must configure the GRE interfaces as core-facing interfaces, and they must be access or trunk interfaces. To configure the bridge domain family on `gr-` interfaces, include the `family bridge` statement at the `[edit interfaces gr-`*`fpc/pic/port`*` unit `*`logical-unit-number`*`]` hierarchy level. To associate the `gr-` interface with a bridge domain, include the `interface gr-`*`fpc/pic/port`* statement at the `[edit routing-instances `*`routing-instance-name`*` bridge-domains `*`bridge-domain-name`*`]` hierarchy level.

You can associate GRE interfaces in a bridge domain with the corresponding VLAN ID or list of VLAN IDs in a bridge domain by including the `vlan-id (all | none | number)` statement or the `vlan-id-list [ vlan-id-numbers ]` statement at the `[edit bridge-domains bridge-domain-name]` hierarchy level. The VLAN IDs configured for the bridge domain must match with the VLAN IDs that you configure for GRE interfaces by using the `vlan-id (all | none | number)` statement or the `vlan-id-list [ vlan-id-numbers ]` statement at the `[edit interfaces gr-fpc/pic/port unit logical-unit-number]` hierarchy level. You can also configure GRE interfaces within a bridge domain associated with a virtual switch instance. Layer 2 Ethernet packets over GRE tunnels are also supported with the GRE key option. The gre-key match condition allows a user to match against the GRE key field, which is an optional field in GRE encapsulated packets. The key can be matched as a single key value, a range of key values, or both.

## Format of GRE Frames and Processing of GRE Interfaces for Layer 2 Ethernet Packets

The GRE frame contains the outer MAC header, outer IP header, GRE header, original layer 2 frame, and frame checksum (FCS).

In the outer MAC header, the following fields are present:

- The outer destination MAC address is set as the next-hop MAC address
- The outer source MAC address is set as the source address of the MX Series router that functions as the gateway
- The outer VLAN tag information

In the outer IP header, the following fields are contained:

- The outer source address is set as the source address of the MX Series router gateway
- The outer destination address is set as the remote GRE tunnel address
- The outer protocol type is set as 47 (encapsulation type is GRE)
- The VLAN ID configuration within the bridge domain updates the VLAN ID of the original Layer 2 header

The gr-interface supports GRE encapsulation over IPv4 and IPv6, which is supported over Layer 3 over GRE. Support for bridging over GRE enables you to configure bridge domain families on gr- interfaces and also enable integrated routing and bridging (IRB) on gr- interfaces. The device control daemon (dcd) that controls the physical and logical interface processes enables the processing of bridge domain families under the GRE interfaces. The kernel supports IRB to send and receive packets on IRB interfaces.

The Packet Forwarding Engine supports the Layer 2 encapsulation and decapsulation over GRE interfaces. The chassis daemon is responsible for creating the GRE physical interface when an FPC comes online and triggering the deletion of the GRE interfaces when the FPC goes offline. The kernel receives the GRE logical interface that is added over the underlying physical interface and propagates the GRE logical interface to other clients, including the Packet Forwarding Engine to create the Layer 2 over GRE data path in the hardware. In addition, it adds the GRE logical interface into a bridge domain. The Packet Forwarding Engine receives interprocess communication message (IPC) from the kernel and adds the interface into the forwarding plane. The existing MTU size for the GRE interface is increased by 22 bytes for the L2 header addition (6 DMAC + 6 SMAC + 4 CVLAN + 4 SVLAN + 2 EtherType)

## Guidelines for Configuring Layer 2 Ethernet Traffic Over GRE Tunnels

Observe the following guidelines while configuring Layer 2 packets to be transmitted over GRE tunnel interfaces on MX Series routers with MPCs:

- For integrated routing and bridging (IRB) to work, at least one Layer 2 interface must be up and active, and it must be associated with the bridge domain as an IRB interface along with a GRE Layer 2 logical interface. This configuration is required to leverege the existing broadcast infrastructure of Layer 2 with IRB.
- Graceful Routing Engine switchover (GRES) is supported and unified ISSU is not currently supported.
- MAC addresses learned from the GRE networks are learned on the bridge domain interfaces associated with the gr-fpc/pic/port.unit logical interface. The MAC addresses are learned on GRE logical interfaces and the Layer 2 token used for forwarding is the token associated with the GRE interface. Destination MAC lookup yields an L2 token, which causes the next-hop lookup. This next-hop is used to forward the packet.
- The GRE tunnel encapsulation and decapsulation next-hops are enhanced to support this functionality. The GRE tunnel encapsulation next-hop is used to encapsulate the outer IP and GRE headers with the incoming L2 packet. The GRE tunnel decapsulation next-hop is used to decapsulate the outer IP and GRE headers, parse the inner Layer 2 packet, and set the protocol as bridge for further bridge domain properties processing in the Packet Forwarding Engine.
- The following packet flows are supported:
  - As part of Layer 2 packet flows, L2 unicast from L2 to GRE, L2 unicast from GRE to L2, Layer 2 broadcast, unknown unicast, and multicast (L2 BUM) from L2 to GRE, and L2 BUM from GRE to L2 are supported.
  - As part of Layer 3 packet flows, L3 Unicast from L2 to GRE, L3 Unicast from GRE to L2, L3 Multicast from L2 to GRE, L3 Multicast from GRE to L2, and L3 Multicast from Internet to GRE and L2 are supported.

- Support for L2 control protocols is not available.
- At the GRE decapsulation side, packets destined to the tunnel IP are processed and decapsulated by the forwarding plane, and inner L2 packets are processed. MAC learned packets are generated for control plane processing for newly learned MAC entries. However, these entries are throttled for MAC learning.
- 802.1x authentication can be used to validate the individual endpoints and protect them from unauthorrized access.
- With the capability to configure bridge domain families on GRE tunnel interfaces, the maximum number of GRE interfaces supported depends on the maximum number of tunnel devices allocated, where each tunnel device can host upto 4000 logical interfaces. The maximum number of logical tunnel interfaces supported is not changed with the support for Layer 2 GRE tunnels. For example, in a 4x10 MIC on MX960 routers, 8000 tunnel logical interfaces can be created.
- The tunnels are pinned to a specific Packet Forwarding Engine instance.
- Statistical information for GRE Layer 2 tunnels is displayed in the output of the `show interfaces gr-`*`fpc/pic/port`* command.
- Only trunk and access mode configuration is supported for the bridge family of GRE interfaces; subinterface style configuration is not supported.
- You can enable a connection to a traditional Layer 2 network. Connection to a VPLS network is not currently supported. IRB in bridge domains with GRE interfaces is supported.
- Virtual switch instances are supported.
- Configuration of the GRE Key and using it to perform the hash load-balancing at the GRE tunnel-initiated and transit routers is supported.

## Sample Scenarios of Configuring Layer 2 Ethernet Traffic Over GRE Tunnels

You can configure Layer 2 Ethernet services over GRE interfaces (`gr-`*`fpc/pic/port`* to use GRE encapsulation). This topic contains the following sections that illustrate sample network deployments that support Layer 2 packets over GRE tunnel interfaces:

**GRE Tunnels with an MX Series Router as the Gateway in Layer 3**

You can configure an MX Series router as the gateway that contains GRE tunnels configured to connect to legacy switches on the one end and to a Layer 3 network on the other end. The Layer 3 network in turn can be linked with multiple servers on a LAN where the GRE tunnel is terminated from the WAN.

#### GRE Tunnels With an MX Series Router as the Gateway and Aggregator

You can configure an MX Series router as the gateway with GRE tunnels configured and also with aggregation specified. The gateway can be connected to legacy switches on one end of the network and the aggregator can be connected to a top-of-rack (ToR) switch, as a QFX Series device, which handles GRE tunneled packets with load balancing. The ToR switch can be connected, in turn, over a Layer 3 GRE tunnel to several servers in data centers.

#### GRE Tunnels with MX Series Gateways for Enterprise and Data Center Servers

You can configure an MX Series router as the gateway with GRE tunnels configured. Over the Internet, GRE tunnels connect multiple gateways, which are MX routers, to servers in enterprises where the GRE tunnel is terminated from the WAN on one end, and to servers in data centers on the other end.

The following configuration scenarios are supported for Layer 2 Ethernet over GRE tunnels:

- In a Layer 2 Ethernet over GRE with VPLS environment, an MX Series router supports Layer 2 over GRE tunnels (without the MPLS layer) and terminate these tunnels into a VPLS or an routed VLAN interface (RVI) into a L3VPN. The tunnels serve to cross the cable modem termination system (CMTS) and cable modem CM infrastructure transparently, up to the MX Series router that serves as the gateway. Every GRE tunnel terminates over a VLAN interface, a VPLS instance, or an IRB interface.

- In a Layer 2 Ethernet over GRE without VPLS environment, Layer 2 VPN encapsulations are needed for conditions that do not involve these protocols. Certain data center users terminate the other end of GRE tunnels directly on the servers on the LAN, while an MX Series router functions as the gateway router between the WAN and LAN. This type of termination of tunnels enables users to build overlay networks within the data center without having to configure end-user VLANs, IP addresses, and other network parameters on the underlying switches. Such a setup simplifies data center network design and provisioning.

> **NOTE**: Layer 2 over GRE is not supported in ACX2200 router.

## Configuring Layer 2 Services over GRE Logical Interfaces in Bridge Domains

You can configure Layer 2 Ethernet services over GRE interfaces (`gr-`*`fpc/pic/port`* to use GRE encapsulation).

To configure a GRE tunnel interface, associate it in a bridge domain within a virtual-switch instance, and specify the amount of bandwidth reserved for tunnel services traffic:

1. Configure GRE tunnel interface and specify the amount of bandwidth to reserve for tunnel traffic on each Packet Forwarding Engine.

```
[edit]
user@host# set chassis fpc slot-number pic slot-number tunnel-services bandwidth (1g | 10g |
20g | 40g)
```

2. Configure the interfaces and their VLAN IDs.

```
[edit]
user@host# set interfaces gr-interface-name unit logical-unit-number family family-name
address address
user@host# set interfaces gr-interface-name unit logical-unit-number family family-name
interface-mode trunk
user@host# set interfaces gr-interface-name unit logical-unit-number family family-name vlan-
id-list vlan-id-list
user@host# set interfaces gr-interface-name unit logical-unit-number tunnel source source-
address
user@host# set interfaces gr-interface-name unit logical-unit-number tunnel destination
destination-address
```

3. Create a virtual switch instance with a bridge domain and associate the GRE logical interfaces.

```
[edit routing-instances]
user@host# set routing-instance-name instance-type virtual-switch
user@host# set routing-instance-name interface interface-name unit logical-unit-number
user@host# set routing-instance-name bridge-domains bridge-domain-name vlan-id vlan-id
```

The VLAN IDs configured for the bridge domain must match with the VLAN IDs that you configure for GRE interfaces by using the `vlan-id (all | none | number)` statement or the `vlan-id-list [ vlan-id-numbers ]` statement at the `[edit interfaces gr-fpc/pic/port unit logical-unit-number]` hierarchy level.

# Example: Configuring Layer 2 Services Over GRE Logical Interfaces in Bridge Domains

**IN THIS SECTION**



This example illustrates how you can configure GRE logical interfaces in a bridge domain. You can also configure a virtual switch instance associated with a bridge domain and include GRE interfaces in the bridge domain. This type of configuration enables you to specify Layer 2 Ethernet packets to be terminated on GRE tunnels. In a Layer 2 Ethernet over GRE with VPLS environment, an MX Series router supports Layer 2 over GRE tunnels (without the MPLS layer) and terminate these tunnels into a VPLS or an routed VLAN interface (RVI) into a L3VPN. The tunnels serve to cross the cable modem termination system (CMTS) and cable modem CM infrastructure transparently, up to the MX Series router that serves as the gateway. Every GRE tunnel terminates over a VLAN interface, a VPLS instance, or an IRB interface.

## Requirements

This example uses the following hardware and software components:

- An MX Series router
- Junos OS Release 15.1R1 or later running on an MX Series router with MPCs.

## Overview

GRE encapsulates packets into IP packets and redirects them to an intermediate host, where they are de-encapsulated and routed to their final destination. Because the route to the intermediate host appears to the inner datagrams as one hop, Juniper Networks EX Series Ethernet switches can operate as if they have a virtual point-to-point connection with each other. GRE tunnels allow routing protocols like RIP and OSPF to forward data packets from one switch to another switch across the Internet. In addition, GRE tunnels can encapsulate multicast data streams for transmission over the Internet.

Ethernet frames have all the essentials for networking, such as globally unique source and destination addresses, error control, and so on. •Ethernet frames can carry any kind of packet. Networking at Layer 2 is protocol independent (independent of the Layer 3 protocol). If more of the end-to-end transfer of information from a source to a destination can be done in the form of Ethernet frames, more of the benefits of Ethernet can be realized on the network. Networking at Layer 2 can be a powerful adjunct to IP networking, but it is not usually a substitute for IP networking.

Consider a sample network topology in which a GRE tunnel interface is configured with the bandwidth set as 10 gigabits per second for tunnel traffic on each Packet Forwarding Engine. The GRE interface, gr-0/1/10.0, is specified with the source address of 192.0.2.2 and the destination address of 192.0.2.1. Two gigabit Ethernet interfaces, ge-0/1/2.0 and ge-0/1/6.0, are also configured. A virtual switch instance, VS1, is defined and a bridge domain, bd0, is associated with VS1. The bridge domain contains the VLAN ID of 10. The GRE interface is configured as a trunk interface and associated with the bridge domain, bd0. With such a setup, Layer 2 Ethernet services can be terminated over GRE tunnel interfaces in virtual switch instances that contain bridge domains.

## Configuration

**IN THIS SECTION**



To configure a GRE tunnel interface, associate it in a bridge domain within a virtual-switch instance, and specify the amount of bandwidth reserved for tunnel services traffic.

**Procedure**

### CLI Quick Configuration

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

```
set chassis fpc 0 pic 1 tunnel-services bandwidth 1g
set chassis network-services enhanced-ip
set interfaces ge-0/1/2 unit 0 family inet address 192.0.2.2/30
set interfaces ge-0/1/6 unit 0 family bridge interface-mode trunk
set interfaces ge-0/1/6 unit 0 family bridge vlan-id-list 1-100
set interfaces gr-0/1/10 unit 0 tunnel source 192.0.2.2
set interfaces gr-0/1/10 unit 0 tunnel destination 192.0.2.1
set interfaces gr-0/1/10 unit 0 family bridge interface-mode trunk
set interfaces gr-0/1/10 unit 0 family bridge vlan-id-list 1-100
```

```
set routing-instances VS1 instance-type virtual-switch
set routing-instances VS1 bridge-domains bd0 vlan-id 10
set routing-instances VS1 interface ge-0/1/6.0
set routing-instances VS1 interface gr-0/1/10.0
```

### Step-by-Step Procedure

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

To configure GRE logical tunnel interfaces for Layer 2 services in bridge domains:

1. Configure GRE tunnel interface and specify the amount of bandwidth to reserve for tunnel traffic on each Packet Forwarding Engine.

```
[edit]
user@host# set chassis fpc 0 pic 1 tunnel-services bandwidth 1g
user@host# set chassis network-services enhanced-ip
```

2. Configure the interfaces and their VLAN IDs.

```
[edit]
user@host# set interfaces ge-0/1/2 unit 0 family inet address 192.0.2.2/30
user@host# set interfaces ge-0/1/6 unit 0 family bridge interface-mode trunk
user@host# set interfaces ge-0/1/6 unit 0 family bridge vlan-id-list 1-100
user@host# set interfaces gr-0/1/10 unit 0 tunnel source 192.0.2.2
user@host# set interfaces gr-0/1/10 unit 0 tunnel destination 192.0.2.1
user@host# set interfaces gr-0/1/10 unit 0 family bridge interface-mode trunk
user@host# set interfaces gr-0/1/10 unit 0 family bridge vlan-id-list 1-100
```

3. Configure the bridge domain in a virtual switch instance and associate the GRE interface with it.

```
[edit]
user@host# set routing-instances VS1 instance-type virtual-switch
user@host# set routing-instances VS1 bridge-domains bd0 vlan-id 10
user@host# set routing-instances VS1 interface ge-0/1/6.0
user@host# set routing-instances VS1 interface gr-0/1/10.0
```

### Results

Display the results of the configuration:

```
user@host> show configuration

chassis {
    fpc 0 {
        pic 1 {
            tunnel-services {
                bandwidth 1g;
            }
        }
    }
    network-services enhanced-ip;
}
interfaces {
    ge-0/1/2 {
        unit 0 {
            family inet {
                address 192.0.2.2/30;
            }
        }
    }
    ge-0/1/6 {
        unit 0 {
            family bridge {
                interface-mode trunk;
                vlan-id-list 1-100;
            }
        }
    }
    gr-0/1/10 {
        unit 0 {
            tunnel {
                source 192.0.2.2;
                destination 192.0.2.1;
            }
            family bridge {
                interface-mode trunk;
                vlan-id-list 1-100;
            }
```

```
        }
    }
}

VS-1 {
    instance-type virtual-switch;
    interface ge-0/1/6.0;
    interface gr-0/1/10.0;
    bridge-domains {
        bd0 {
        vlan-id 10;
        }
    }
}
```

## Verification

**IN THIS SECTION**



To confirm that the configuration is working properly, perform these tasks:

### Verifying the MAC Addresses Learned on GRE Interfaces

#### Purpose

Display the MAC addresses learned on a GRE logical interface.

#### Action

From operational mode, use the `show bridge mac-table` command

```
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)
```

```
Routing instance : default-switch
 Bridging domain : vlan-1, VLAN : 1
   MAC                 MAC      Logical
   address             flags    interface
   00:00:5e:00:53:f7   D,SE     gr-1/2/10.0
   00:00:5e:00:53:32   D,SE     gr-1/2/10.0
   00:00:5e:00:53:21   DL       ge-1/0/0.0
   00:00:5e:00:53:11   DL       ge-1/1/0.0


Routing instance : default-switch
 Bridging domain : vlan-2, VLAN : 2
   MAC                 MAC      Logical
   address             flags    interface
   00:00:5e:00:53:33   D,SE     gr-1/2/10.1
   00:00:5e:00:53:10   DL       ge-1/0/0.1
   00:00:5e:00:53:23   DL       ge-1/1/0.1
```

## Meaning

The output displays MAC addresses learned on GRE logical tunnels.

### Verifying the MAC Address Learning Status

## Purpose

Display the status of MAC address learning properties in the MAC address and MAC flags fields.

## Action

From operational mode, enter the `show vpls mac-table` command.

```
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : vpls_4site:1000
 Bridging domain : __vpls_4site:1000__,   MAC                 MAC      Logical
   address             flags    interface
   00:00:5e:00:53:f4   D,SE     ge-4/2/0.1000
   00:00:5e:00:53:02   D,SE     lsi.1052004
   00:00:5e:00:53:03   D,SE     lsi.1048840
```

```
   00:00:5e:00:53:04   D,SE     lsi.1052005
   00:00:5e:00:53:33   D,SE     gr-1/2/10.10


user@host> show interfaces gr-2/2/10
Physical interface: gr-2/2/10, Enabled, Physical link is Up
  Interface index: 214, SNMP ifIndex: 690
  Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 1000mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)

  Logical interface gr-2/2/10.0 (Index 342) (SNMP ifIndex 10834)
    Flags: Up Point-To-Point SNMP-Traps 0x4000 IP-Header
198.51.100.1:198.51.100.254:47:df:64:0000000000000000 Encapsulation: GRE-NULL
    L2 Routing Instance: vs1, L3 Routing Instance: default
    Copy-tos-to-outer-ip-header: Off
    Gre keepalives configured: Off, Gre keepalives adjacency state: down
    Input packets : 2
    Output packets: 0
    Protocol bridge, MTU: 1476
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 6/8, Local: 6.0.0.1, Broadcast: 6.255.255.255

user@host> show interfaces gr-2/2/10.0
  Logical interface gr-2/2/10.0 (Index 342) (SNMP ifIndex 10834)
    Flags: Up Point-To-Point SNMP-Traps 0x4000 IP-Header
198.51.100.1:198.51.100.254:47:df:64:0000000000000000 Encapsulation: GRE-NULL
   L2 Routing Instance: vs1, L3 Routing Instance: default
    Copy-tos-to-outer-ip-header: Off
    Gre keepalives configured: Off, Gre keepalives adjacency state: down
    Input packets : 2
    Output packets: 0
    Protocol bridge, MTU: 1476
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 6/8, Local: 6.0.0.1, Broadcast: 6.255.255.255
```

### Meaning

The output displays the status of MAC address learning properties in the MAC address and MAC flags fields. The output displays the names of the routing instances associated with the GRE interfaces are displayed.

## Example: Configuring Layer 2 Services Over GRE Logical Interfaces in Bridge Domains with IPv6 Transport

IN THIS SECTION



This example illustrates how you can configure GRE logical interfaces in a bridge domain. You can also configure a virtual switch instance associated with a bridge domain and include GRE interfaces in the bridge domain. This type of configuration enables you to specify Layer 2 Ethernet packets to be terminated on GRE tunnels. In a Layer 2 Ethernet over GRE with VPLS environment, an MX Series router supports Layer 2 over GRE tunnels (without the MPLS layer) and terminate these tunnels into a VPLS or an routed VLAN interface (RVI) into a L3VPN. The tunnels serve to cross the cable modem termination system (CMTS) and cable modem CM infrastructure transparently, up to the MX Series router that serves as the gateway. Every GRE tunnel terminates over a VLAN interface, a VPLS instance, or an IRB interface.

### Requirements

This example uses the following hardware and software components:

- Two MX Series routers
- Junos OS Release 19.R1 or later running on MX Series routers with MPCs.

### Overview

GRE encapsulated IPv6 packets are redirected to an intermediate host where GRE header is decapsulated and routed to the IPv6 destination.

Consider a sample network topology with two devices. On device 1, GRE tunnel interface is configured with the bandwidth set as 1 gigabits per second for tunnel traffic on each Packet Forwarding Engine. The GRE interface, gr-0/0/10.0, is specified with the source address of 2001:DB8::2:1 and the destination address of 2001:DB8::3:1. Two interfaces, ae0 and xe-0/0/19, are also configured. A virtual switch instance, VS1, is defined and a bridge domain, bd1, is associated with VS1. The bridge domain contains the VLAN ID of 20. The GRE interface is configured as a trunk interface and associated with the bridge domain, bd1. With such a setup, Layer 2 Ethernet services can be terminated over GRE tunnel interfaces in virtual switch instances that contain bridge domains.

On device 2, GRE tunnel interface is configured with the bandwidth set as 1 gigabits per second for tunnel traffic on each Packet Forwarding Engine. The GRE interface, gr-0/0/10.0, is specified with the source address of 2001:DB8::21:1 and the destination address of 2001:DB8::31:1. Two interfaces, ae0 and xe-0/0/1, are also configured. A virtual switch instance, VS1, is defined and a bridge domain, bd1, is associated with VS1. The bridge domain contains the VLAN ID of 20. The GRE interface is configured as an access interface and associated with the bridge domain, bd1.

## Configuration

**IN THIS SECTION**



To configure a GRE tunnel interface, associate it in a bridge domain within a virtual-switch instance, and specify the amount of bandwidth reserved for tunnel services traffic.

**Procedure**

### CLI Quick Configuration

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

For Device 1:

```
set chassis aggregated-devices ethernet device-count 2
set chassis fpc 0 pic 0 tunnel-services bandwidth 1g
set chassis network-services enhanced-ip
set interfaces ae0 unit 0 family inet6 address 2001:DB8::1:1/32;
set interfaces xe-0/0/19 unit 0 family bridge interface-mode trunk
```

```
set interfaces xe-0/0/19 unit 0 family bridge vlan-id-list 20-21
set interfaces xe-1/0/2 gigether-options 802.3ad ae0
set interfaces xe-1/0/3 gigether-options 802.3ad ae0
set interfaces gr-0/0/10.0 unit 0 tunnel source 2001:DB8::2:1
set interfaces gr-0/0/10.0 unit 0 tunnel destination 2001:DB8::3:1/32
set interfaces gr-0/0/10.0 unit 0 family bridge interface-mode trunk
set interfaces gr-0/0/10.0 unit 0 family bridge vlan-id-list 20-30
set routing-instances VS1 instance-type virtual-switch
set routing-instances VS1 bridge-domains bd1 vlan-id 20
set routing-instances VS1 interface xe-0/0/19.0
set routing-instances VS1 interface gr-0/0/10.0
```

For device 2:

```
set chassis aggregated-devices ethernet device-count 2
set chassis fpc 0 pic 0 tunnel-services bandwidth 1g
set chassis network-services enhanced-ip
set interfaces ae0 unit 0 family inet6 address 2001:DB8::11:1/32;
set interfaces xe-0/0/1unit 0 family bridge interface-mode trunk
set interfaces xe-0/0/1 unit 0 family bridge vlan-id-list 20-21
set interfaces xe-1/0/2 gigether-options 802.3ad ae0
set interfaces xe-1/0/3 gigether-options 802.3ad ae0
set interfaces gr-0/0/10.0  unit 0 tunnel source 2001:DB8::21:1
set interfaces gr-0/0/10.0  unit 0 tunnel destination 2001:DB8::31:1/32
set interfaces gr-0/0/10.0  unit 0 family bridge interface-mode access
set interfaces gr-0/0/10.0  unit 0 family bridge vlan-id-list 20-30
set routing-instances VS1 instance-type virtual-switch
set routing-instances VS1 bridge-domains bd1 vlan-id 20
set routing-instances VS1 interface xe-0/0/1.0
set routing-instances VS1 interface gr-0/0/10.0
```

### Step-by-Step Procedure

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

To configure GRE logical tunnel interfaces over IPv6 for Layer 2 services in bridge domains for Device1 and Device2:

1. Configure GRE tunnel interface and specify the amount of bandwidth to reserve for tunnel traffic on each Packet Forwarding Engine of Device1.

```
[edit]
user@host# set chassis aggregated-devices ethernet device-count 2
user@host# set chassis fpc 0 pic 0 tunnel-services bandwidth 1g
user@host# set chassis network-services enhanced-ip
```

2. Configure the interfaces and their VLAN IDs.

```
[edit]
user@host# set interfaces ae0 unit 0 family inet6 address 2001:DB8::1:1/32;
user@host# set interfaces xe-0/0/19.0 unit 0 family bridge interface-mode trunk
user@host# set interfaces xe-0/0/19.0 unit 0 family bridge vlan-id-list 20-21
user@host# set interfaces xe-1/0/2 gigether-options 802.3ad ae0
user@host# set interfaces xe-1/0/3 gigether-options 802.3ad ae0
user@host# set interfaces gr-0/0/10.0  unit 0 tunnel source 2001:DB8::2:1
user@host# set interfaces gr-0/0/10.0  unit 0 tunnel destination 2001:DB8::3:1
user@host# set interfaces gr-0/0/10.0  unit 0 family bridge interface-mode trunk
user@host# set interfaces gr-0/0/10.0  unit 0 family bridge vlan-id-list 20-30
```

3. Configure the bridge domain in a virtual switch instance and associate the GRE interface with it.

```
[edit]
user@host# set routing-instances VS1 instance-type virtual-switch
user@host# set routing-instances VS1 bridge-domains bd1 vlan-id 20
user@host# set routing-instances VS1 interface xe-0/0/19.0
user@host# set routing-instances VS1 interface gr-0/0/10.0
```

4. Configure GRE tunnel interface and specify the amount of bandwidth to reserve for tunnel traffic on each Packet Forwarding Engine of Device2.

```
[edit]
user@host# set chassis aggregated-devices ethernet device-count 2
user@host# set chassis fpc 0 pic 0 tunnel-services bandwidth 1g
user@host# set chassis network-services enhanced-ip
```

5. Configure the interfaces and their VLAN IDs.

```
[edit]
user@host# set interfaces ae0 unit 0 family inet6 address 2001:DB8::11:1/32;
user@host# set interfaces xe-0/0/1unit 0 family bridge interface-mode trunk
user@host# set interfaces xe-0/0/1 unit 0 family bridge vlan-id-list 20-21
user@host# set interfaces xe-1/0/2 gigether-options 802.3ad ae0
user@host# set interfaces xe-1/0/3 gigether-options 802.3ad ae0
user@host# set interfaces gr-0/0/10.0  unit 0 tunnel source 2001:DB8::21:1
user@host# set interfaces gr-0/0/10.0  unit 0 tunnel destination 2001:DB8::31:1/32
user@host# set interfaces gr-0/0/10.0  unit 0 family bridge interface-mode trunk
user@host# set interfaces gr-0/0/10.0  unit 0 family bridge vlan-id-list 20-30
```

6. Configure the bridge domain in a virtual switch instance and associate the GRE interface with it.

```
[edit]
user@host# set routing-instances VS1 instance-type virtual-switch
user@host# set routing-instances VS1 bridge-domains bd1 vlan-id 20
user@host# set routing-instances VS1 bridge-domains routing-interface irb.0
user@host# set routing-instances VS1 interface xe-0/0/1.0
user@host# set routing-instances VS1 interface gr-0/0/10.0
```

### Results

Display the results of the configuration on Device 1:

```
user@host> show configuration
chassis {
    fpc 0 {
        pic 0 {
            tunnel-services {
                bandwidth 1g;
            }
        }
    }
    network-services enhanced-ip;
}
interfaces {
    ae0 {
        unit 0 {
```

```
            family inet6 {
                address 2001:DB8::1:1/32;
            }
        }
    }
    xe-0/0/19 {
        unit 0 {
            family bridge {
                interface-mode trunk;
                vlan-id-list 20-21;
            }
        }
    }
    gr-0/0/10 {
        unit 0 {
            tunnel {
                source 2001:DB8::2:1;
                destination 2001:DB8::3:1;
            }
            family bridge {
                interface-mode trunk;
                vlan-id-list 20-30;
            }
        }
    }
}

VS-1 {
    instance-type virtual-switch;
    interface xe-0/0/19.0;
    interface gr-0/0/10.0;
    bridge-domains {
        bd1 {
        vlan-id 20;
        }
    }
}
```

Display the results of the configuration on Device 2:

```
user@host> show configuration
chassis {
```

```
    fpc 0 {
        pic 0 {
            tunnel-services {
                bandwidth 1g;
            }
        }
    }
    network-services enhanced-ip;
}
interfaces {
    ae0 {
        unit 0 {
            family inet6 {
                address 2001:DB8::11:1/32;
            }
        }
    }
    xe-0/0/1 {
        unit 0 {
            family bridge {
                interface-mode trunk;
                vlan-id-list 20-21;
            }
        }
    }
    gr-0/0/10 {
        unit 0 {
            tunnel {
                source 2001:DB8::21:1;
                destination 2001:DB8::31:1;
            }
            family bridge {
                interface-mode access;
                vlan-id-list 20-30;
            }
        }
    }
}

VS-1 {
    instance-type virtual-switch;
    interface xe-0/0/1.0;
    interface gr-0/0/10.0;
```

```
    bridge-domains {
        bd1 {
        vlan-id 20;
        }
    }
}
```

## Verification

**IN THIS SECTION**



To confirm that the configuration is working properly, perform these tasks:

#### Verifying the MAC Addresses Learned on GRE Interfaces

### Purpose

Display the MAC addresses learned on a GRE logical interface.

### Action

From operational mode, use the `show bridge mac-table` command

```
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : VS1
Bridging domain : bd1, VLAN : 20
   MAC                   MAC      Logical          NH     RTR
   address               flags    interface        Index  ID
   00:00:00:11:11:11     D        gr-0/0/10.0
   00:00:00:11:11:12     D        gr-0/0/10.0
   00:00:00:11:11:13     D        gr-0/0/10.0
   00:00:00:11:11:14     D        gr-0/0/10.0
   00:00:00:11:11:15     D        gr-0/0/10.0
   00:00:00:11:11:16     D        gr-0/0/10.0
```

```
00:00:00:11:11:17   D        gr-0/0/10.0
00:00:00:11:11:18   D        gr-0/0/10.0
00:00:00:11:11:19   D        gr-0/0/10.0
00:00:00:11:11:1a   D        gr-0/0/10.0
00:00:00:22:22:22   D        xe-0/0/19.0
00:00:00:22:22:23   D        xe-0/0/19.0
00:00:00:22:22:24   D        xe-0/0/19.0
00:00:00:22:22:25   D        xe-0/0/19.0
00:00:00:22:22:26   D        xe-0/0/19.0
00:00:00:22:22:27   D        xe-0/0/19.0
00:00:00:22:22:28   D        xe-0/0/19.0
00:00:00:22:22:29   D        xe-0/0/19.0
00:00:00:22:22:2a   D        xe-0/0/19.0
00:00:00:22:22:2b   D        xe-0/0/19.0
```

**Meaning**

The output displays MAC addresses learned on GRE logical tunnels.

# Configuring PIM Tunnels

PIM tunnels are enabled automatically on routers that have a tunnel PIC and on which you enable PIM sparse mode. You do not need to configure the tunnel interface.

PIM tunnels are unidirectional.

In PIM sparse mode, the first-hop router encapsulates packets destined for the rendezvous point (RP) router. The packets are encapsulated with a unicast header and are forwarded through a unicast tunnel to the RP. The RP then de-encapsulates the packets and transmits them through its multicast tree. To perform the encapsulation and de-encapsulation, the first-hop and RP routers must be equipped with Tunnel PICs.

The Junos OS creates two interfaces to handle PIM tunnels:

- `pe`—Encapsulates packets destined for the RP. This interface is present on the first-hop router.
- `pd`—De-encapsulates packets at the RP. This interface is present on the RP.

**NOTE**: The pe and pd interfaces do not support class-of-service (CoS) configurations.

RELATED DOCUMENTATION



# Facilitating VRF Table Lookup Using Virtual Loopback Tunnel Interfaces

IN THIS SECTION



## Configuring Virtual Loopback Tunnels for VRF Table Lookup

To enable egress filtering, you can either configure filtering based on the IP header, or you can configure a virtual loopback tunnel on routers equipped with a Tunnel PIC. Table 10 on page 134 describes each method.

**Table 10: Methods for Configuring Egress Filtering**

| Method | Interface Type | Configuration Guidelines | Comments |
|---|---|---|---|
| Filter traffic based on the IP header | Nonchannelized Point-to-Point Protocol / High Level Data Link Control (PPP/ HDLC) core-facing SONET/SDH interfaces | Include the `vrf-table-label` statement at the `[edit routing-instances instance-name]` hierarchy level.<br><br>For more information, see the Junos OS VPNs Library for Routing Devices. | There is no restriction on customer-edge (CE) router-to-provider edge (PE) router interfaces. |
| Configure a virtual loopback tunnel on routers equipped with a Tunnel PIC | All interfaces | See the guidelines in this section. | Router must be equipped with a Tunnel PIC.<br><br>There is no restriction on the type of core-facing interface used or CE router-to-PE router interface used.<br><br>You cannot configure a virtual loopback tunnel and the `vrf-table-label` statement at the same time. |

You can configure a virtual loopback tunnel to facilitate VRF table lookup based on MPLS labels. You might want to enable this functionality so you can do either of the following:

- Forward traffic on a PE router to CE device interface, in a shared medium, where the CE device is a Layer 2 switch without IP capabilities (for example, a metro Ethernet switch).

  The first lookup is done based on the VPN label to determine which VRF table to refer to, and the second lookup is done on the IP header to determine how to forward packets to the correct end hosts on the shared medium.

- Perform egress filtering at the egress PE router.

  The first lookup on the VPN label is done to determine which VRF table to refer to, and the second lookup is done on the IP header to determine how to filter and forward packets. You can enable this functionality by configuring output filters on the VRF interfaces.

To configure a virtual loopback tunnel to facilitate VRF table lookup based on MPLS labels, you specify a virtual loopback tunnel interface name and associate it with a routing instance that belongs to a particular routing table. The packet loops back through the virtual loopback tunnel for route lookup. To specify a virtual loopback tunnel interface name, you configure the virtual loopback tunnel interface at the `[edit interfaces]` hierarchy level and include the `family inet` and `family mpls` statements:

```
vt-fpc/pic/port {
    unit 0 {
        family inet;
        family mpls;
    }
    unit 1 {
        family inet;
    }
}
```

To associate the virtual loopback tunnel with a routing instance, include the virtual loopback tunnel interface name at the `[edit routing-instances]` hierarchy level:

```
interface vt-fpc/pic/port;
```

**NOTE**: On virtual loopback tunnel interfaces, none of the logical interface statements except the `family` statement is supported. Note that you can configure only `inet` and `mpls` families, and you cannot configure IPv4 or IPv6 addresses on virtual loopback tunnel interfaces. Also, virtual loopback tunnel interfaces do not support class-of-service (CoS) configurations.

SEE ALSO

Tunnel Services Overview | 2

## Configuring Tunnel Interfaces for Routing Table Lookup

To configure tunnel interfaces to facilitate routing table lookups for VPNs, you specify a tunnel's endpoint IP addresses and associate them with a routing instance that belongs to a particular routing table. This enables the Junos OS to search in the appropriate routing table for the route prefix, because the same prefix can appear in multiple routing tables. To configure the destination VPN, include the `routing-instance` statement:

```
routing-instance {
    destination routing-instance-name;
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces gr-`*fpc/pic/port*` unit `*logical-unit-number*` tunnel]`
- `[edit logical-systems `*logical-system-name*` interfaces gr-`*fpc/pic/port*` unit `*logical-unit-number*` tunnel]`

This configuration indicates that the tunnel's destination address is in routing instance *routing-instance-name*. By default, the tunnel route prefixes are assumed to be in the default Internet routing table `inet.0`.

> **NOTE**: If you configure a virtual loopback tunnel interface and the `vrf-table-label` statement on the same routing instance, the `vrf-table-label` statement takes precedence over the virtual loopback tunnel interface. For more information, see "Configuring Virtual Loopback Tunnels for VRF Table Lookup" on page 133.

For more information about VPNs, see the Junos OS VPNs Library for Routing Devices.

SEE ALSO

Tunnel Services Overview | 2
destination (Routing Instance) | 181

## Example: Configuring a Virtual Loopback Tunnel for VRF Table Lookup

Configure a virtual loopback tunnel for VRF table lookup:

```
[edit routing-instances]
routing-instance-1 {
    instance-type vrf;
    interface vt-1/0/0.0;
    interface so-0/2/2.0;
```

```
    route-distinguisher 2:3;
    vrf-import VPN-A-import;
    vrf-export VPN-A-export;
    routing-options {
        static {
            route 10.0.0.0/8 next-hop so-0/2/2.0;
        }
    }
}
routing-instance-2 {
    instance-type vrf;
    interface vt-1/0/0.1;
    interface so-0/3/2.0;
    route-distinguisher 4:5;
    vrf-import VPN-B-import;
    vrf-export VPN-B-export;
    routing-options {
        static {
            route 10.0.0.0/8 next-hop so-0/3/2.0;
        }
    }
}
[edit interfaces]
vt-1/0/0 {
    unit 0 {
        family inet;
        family mpls;
    }
    unit 1 {
        family inet;
    }
}
```

#### SEE ALSO

Tunnel Services Overview | 2

## Example: Virtual Routing and Forwarding (VRF) and Service Configuration

The following example combines virtual routing and forwarding (*VRF*) and services configuration:

```
[edit policy-options]
policy-statement test-policy {
    term t1 {
        then reject;
    }
}
[edit routing-instances]
test {
    interface ge-0/2/0.0;
    interface sp-1/3/0.20;
    instance-type vrf;
    route-distinguisher 10.58.255.1:37;
    vrf-import test-policy;
    vrf-export test-policy;
    routing-options {
        static {
            route 0.0.0.0/0 next-table inet.0;
        }
    }
}
[edit interfaces]
ge-0/2/0 {
    unit 0 {
        family inet {
            service {
                input service-set nat-me;
                output service-set nat-me;
            }
        }
    }
}
sp-1/3/0 {
    unit 0 {
        family inet;
    }
    unit 20 {
```

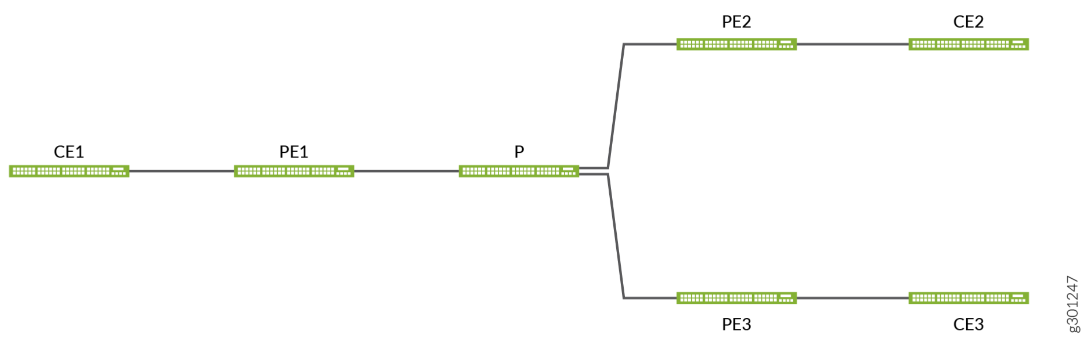
```
        family inet;
        service-domain inside;
    }
    unit 21 {
        family inet;
        service-domain outside;
    }
    [edit services]
    stateful-firewall {
        rule allow-any-input {
            match-direction input;
            term t1 {
                then accept;
            }
        }
    }
    nat {
        pool hide-pool {
            address 10.58.16.100;
            port automatic;
        }
        rule hide-all-input {
            match-direction input;
            term t1 {
                then {
                    translated {
                        source-pool hide-pool;
                        translation-type source napt-44;
                    }
                }
            }
        }
    }
    service-set nat-me {
        stateful-firewall-rules allow-any-input;
        nat-rules hide-all-input;
        interface-service {
            service-interface sp-1/3/0.20;
        }
    }
}
```

# BGP Layer 3 VPN over IP-IP Tunnels Overview

The traditional VPN services use the label-based forwarding technique of MPLS. Some networks might transition from MPLS network to IP fabric core network. VPN label is not supported in IP fabric core network.

We introduce support for BGP Layer 3 VPN over IP over IP (IP-IP) tunnels to create a new transport service that does not require a VPN label to identify the VRF at Egress PE. This offer carrying different types of traffic using the same infrastructure and same BGP topology with RD and route targets over the IP-IP network. IP-IP tunnels terminate into service-layer VRF, so you do not need to use a service label. This feature allows interoperability between the new VRF and traditional VRF, so both types of overlays can coexist in your network. You can use this feature to transition from an MPLS network to an IP fabric core network and to protect your network from distributed denial-of-service (DDoS) attacks.

**Figure 6: Sample Network That Shows Coexistence of New and Traditional VRFs**




In Figure 6 on page 140, PE1 is an Egress PE that supports both traditional and new types of tunnels, so it will advertise both these types of tunnels in its L3VPN routes. PE2 is an Ingress PE that uses new type of tunnel to reach CE1 and PE3 is an Ingress PE that uses traditional tunnel to reach CE1. If an IP-IP tunnel is selected to forward the traffic, the application service label is ignored and a firewall filter is used to demultiplex the traffic. L3VPN traffic is decapsulated and then do route lookup in the VRF while IPv4/IPv6 traffic is decapsulated and do route lookup in the inet.0/inet6.0 table.

Threat Mitigation system prefixes are exchanged via BGP inetvpn or inet6vpn unicast updates. These BGP updates carry tunnel encapsulate attributes. The BGP L3VPN separates the threat mitigation prefix from the internet routes and does not impact the internet route’s best path selection, thus minimizing the probability of forming a routing loop. In Junos OS 20.3R1 and later releases, you can choose to use VPN over MPLS transport or switch to IP over IP tunnel according to the tunnel attribute. If the receiver router is running on Junos OS 20.2 or previous releases, the path attribute is ignored and uses the traditional MPLS transport service.

To use VPN over an IP-IP tunnel, you need to configure a tunnel-attribute. When routes are advertised directly from the VRF table, you can use a BGP or VRF export policy to attach the tunnel attribute. When the advertise-from-main-vpn-tables statement is enabled, or when the device acts as a route reflector or an AS boundary router, you must attach the tunnel attribute using a BGP export policy under BGP.

You can configure the receiver to program the dynamic tunnel using the tunnel attribute to enable tunnel-attribute on trusted BGP peers. The route reflector is able to reflect the route with the tunnel attribute even if this is not configured.

2

CHAPTER

# Encryption Services

# Encryption Services Overview

The IP Security (IPsec) architecture provides a security suite for the IP version 4 (IPv4) and IP version 6 (IPv6) network layers. The suite provides functionality such as authentication of origin, data integrity, confidentiality, replay protection, and nonrepudiation of source. It also defines mechanisms for key generation and exchange, management of security associations, and support for digital certificates.

IPsec defines a security association (SA) and key management framework that can be used with any network layer protocol. The SA specifies what protection policy to apply to traffic between two IP-layer entities. For more information, see the Junos OS Administration Library for Routing Devices. The standards are defined in the following RFCs:

- RFC 2401, *Security Architecture for the Internet Protocol*
- RFC 2406, *IP Encapsulating Security Payload (ESP)*

# Configuring Encryption Interfaces

**IN THIS SECTION**

# Configuring Encryption Interfaces

**IN THIS SECTION**



When you configure the encryption interface, you associate the configured SA with a logical interface. This configuration defines the tunnel, including the logical unit, tunnel addresses, maximum transmission unit (MTU), optional interface addresses, and the name of the IPsec SA to apply to traffic. To configure an encryption interface, include the following statements at the `[edit interfaces es-`*`fpc/pic/port`* `unit` *`logical-unit-number`*`]` hierarchy level:

```
family inet {
    ipsec-sa ipsec-sa; # name of security association to apply to packet
    address address;                 # local interface address inside local VPN
    destination address;                 # destination address inside remote VPN
}
tunnel {
    source source-address;
    destination destination-address;
}
```

The addresses configured as the tunnel source and destination are the addresses in the outer IP header of the tunnel.

**NOTE**: You must configure the tunnel source address locally on the router, and the tunnel destination address must be a valid address for the security gateway terminating the tunnel.

The ES Physical Interface Card (PIC) is supported on M Series and T Series routers.

The SA must be a valid tunnel-mode SA. The interface address and destination address listed are optional. The destination address allows the user to configure a static route to encrypt traffic. If a static route uses that destination address as the next hop, traffic is forwarded through the portion of the tunnel in which encryption occurs.

### Specifying the Security Association Name for Encryption Interfaces

The security association is the set of properties that defines the protocols for encrypting Internet traffic. To configure encryption interfaces, you specify the SA name associated with the interface by including the `ipsec-sa` statement at the `[edit interfaces es-fpc/pic/port unit logical-unit-number family inet]` hierarchy level:

```
ipsec-sa sa-name;
```

For information about configuring the security association, see "Configuring Filters for Traffic Transiting the ES PIC" on page 146.

### Configuring the MTU for Encryption Interfaces

The protocol MTU value for encryption interfaces must always be less than the default interface MTU value of 3900 bytes; the configuration fails to commit if you select a greater value. To set the MTU value, include the `mtu` statement at the `[edit interfaces interface-name unit logical-unit-number family inet]` hierarchy level:

```
mtu bytes;
```

For more information, see the Junos OS Network Interfaces Library for Routing Devices.

### Example: Configuring an Encryption Interface

Configure an IPsec tunnel as a logical interface on the ES PIC. The logical interface specifies the tunnel through which the encrypted traffic travels. The `ipsec-sa` statement associates the security profile with the interface.

```
[edit interfaces]
es-0/0/0 {
    unit 0 {
        tunnel {
            source 10.5.5.5;                      # tunnel source address
            destination 10.6.6.6;                         # tunnel destination address
        }
        family inet {
            ipsec-sa manual-sa1; # name of security association to apply to packet
            mtu 3800;
            address 10.1.1.8/32 { # local interface address inside local VPN
```

```
            destination 10.2.2.254;  # destination address inside remote VPN
        }
    }
}
```

# Configuring Filters for Traffic Transiting the ES PIC

IN THIS SECTION



This section contains the following topics:

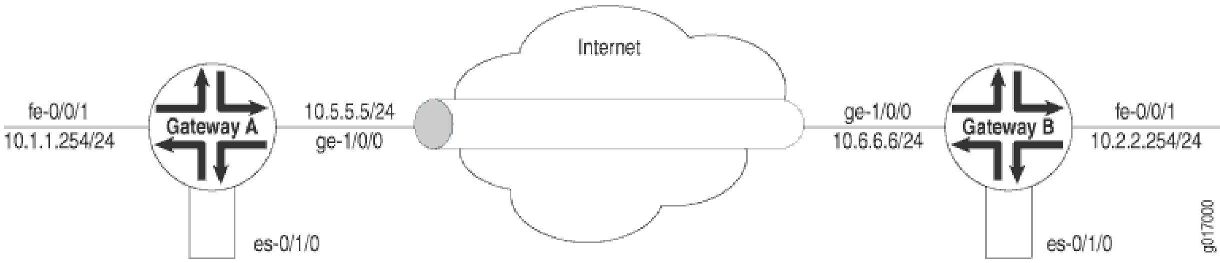
## Traffic Overview

Traffic configuration defines the traffic that must flow through the tunnel. You configure outbound and inbound firewall filters, which identify and direct traffic to be encrypted and confirm that decrypted traffic parameters match those defined for the given tunnel. The outbound filter is applied to the LAN or WAN interface for the incoming traffic you want to encrypt. The inbound filter is applied to the ES PIC to check the policy for traffic coming in from the remote host. Because of the complexity of configuring a router to forward packets, no automatic checking is done to ensure that the configuration is correct.

**NOTE**: The valid firewall filters statements for IPsec are `destination-port`, `source-port`, `protocol`, `destination-address`, and `source-address`.

In Figure 7 on page 147, Gateway A protects the network `10.1.1.0/24`, and Gateway B protects the network `10.2.2.0/24`. The gateways are connected by an IPsec tunnel. For more information about firewalls, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

**Figure 7: Example: IPsec Tunnel Connecting Security Gateways**




The SA and ES interface for security Gateway A are configured as follows:

```
[edit security ipsec]
security-association manual-sa1 {
    manual {
        direction bidirectional {
            protocol esp;
            spi 2312;
            authentication {
                algorithm hmac-md5-96;
                key ascii-text 1234123412341234;
            }
            encryption {
                algorithm 3des-cbc;
                key ascii-text 123456789009876543211234;
            }
        }
    }
}
[edit interfaces es-0/1/0]
unit 0 {
    tunnel {
        source 10.5.5.5;
        destination 10.6.6.6;
    }
    family inet {
        ipsec-sa manual-sa1;
        address 10.1.1.8/32 {
            destination 10.2.2.254;
        }
    }
}
```

## Configuring the Security Association

To configure the SA, include the `security-association` statement at the `[edit security]` hierarchy level:

```
security-association name {
    mode (tunnel | transport);
    manual {
        direction (inbound | outbound | bi-directional) {
            auxiliary-spi auxiliary-spi-value;
            spi spi-value;
            protocol (ah | esp | bundle);
            authentication {
                algorithm (hmac-md5-96 | hmac-sha1-96);
                key (ascii-text key | hexadecimal key);
            }
            encryption {
                algorithm (des-cbc | 3des-cbc);
                key (ascii-text key | hexadecimal key);
            }
        }
        dynamic {
            replay-window-size (32 | 64);
            ipsec-policy policy-name;
        }
    }
}
```

For more information about configuring an SA, see the Junos OS Administration Library for Routing Devices. For information about applying the SA to an interface, see 147531"Specifying the Security Association Name for Encryption Interfaces" on page 144.

## Configuring an Outbound Traffic Filter

To configure the outbound traffic filter, include the `filter` statement at the `[edit firewall]` hierarchy level:

```
filter filter-name {
    term term-name {
        from {
            match-conditions;
        }
        then {
```

```
            action;
            action-modifiers;
        }
    }
}
```

For more information, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

### Example: Configuring an Outbound Traffic Filter

Firewall filters for outbound traffic direct the traffic through the desired IPsec tunnel and ensure that the tunneled traffic goes out the appropriate interface (see Figure 7 on page 147). Here, an outbound firewall filter is created on security Gateway A; it identifies the traffic to be encrypted and adds it to the input side of the interface that carries the internal virtual private network (VPN) traffic:

```
[edit firewall]
filter ipsec-encrypt-policy-filter {
    term term1 {
        from {
            source-address {          # local network
            10.1.1.0/24;
        }
        destination-address {     # remote network
        10.2.2.0/24;
    }
}
then ipsec-sa manual-sa1;     # apply SA name to packet
term default {
    then accept;
}
```

**NOTE**: The source address, port, and protocol on the outbound traffic filter must match the destination address, port, and protocol on the inbound traffic filter. The destination address, port, and protocol on the outbound traffic filter must match the source address, port, and protocol on the inbound traffic filter.

## Applying the Outbound Traffic Filter

After you have configured the outbound firewall filter, you apply it by including the `filter` statement at the `[edit interfaces *interface-name* unit *logical-unit-number* family inet]` hierarchy level:

```
filter {
    input filter-name;
}
```

### Example: Applying the Outbound Traffic Filter

Apply the outbound traffic filter. The outbound filter is applied on the Fast Ethernet interface at the `[edit interfaces fe-0/0/1 unit 0 family inet]` hierarchy level. Any packet matching the IPsec action term (`term 1`) on the input filter (`ipsec-encrypt-policy-filter`), configured on the Fast Ethernet interface, is directed to the ES PIC interface at the `[edit interfaces es-0/1/0 unit 0 family inet]` hierarchy level. So, if a packet arrives from the source address `10.1.1.0/24` and goes to the destination address `10.2.2.0/24`, the Packet Forwarding Engine directs the packet to the ES PIC interface, which is configured with the `manual-sa1` SA. The ES PIC receives the packet, applies the `manual-sa1` SA, and sends the packet through the tunnel.

The router must have a route to the tunnel end point; add a static route if necessary.

```
[edit interfaces]
fe-0/0/1 {
    unit 0 {
        family inet {
            filter {
                input ipsec-encrypt-policy-filter;
            }
            address 10.1.1.254/24;
        }
    }
}
```

## Configuring an Inbound Traffic Filter

To configure an inbound traffic filter, include the `filter` statement at the `[edit firewall]` hierarchy level:

```
filter filter-name {
    term term-name {
```

```
        from {
            match-conditions;
        }
        then {
            action;
            action-modifiers;
        }
    }
}
```

For more information, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

### Example: Configuring an Inbound Traffic Filter

Configure an inbound firewall filter. This filter performs the final IPsec policy check and is created on security gateway A. The policy check ensures that only packets that match the traffic configured for this tunnel are accepted.

```
[edit firewall]
filter ipsec-decrypt-policy-filter {
    term term1 {                                  # perform policy check
    from {
        source-address {                      # remote network
        10.2.2.0/24;
    }
    destination-address {                 # local network
    10.1.1.0/24;
}
then accept;
```

## Applying the Inbound Traffic Filter to the Encryption Interface

After you create the inbound firewall filter, you can apply it to the ES PIC. To apply the filter to the ES PIC, include the `filter` statement at the `[edit interfaces es-`*`fpc/pic/port`*` unit `*`logical-unit-number`*` family inet filter]` hierarchy level:

```
filter {
    input filter;
}
```

The input filter is the name of the filter applied to received traffic. For a configuration example, see "Example: Configuring an Inbound Traffic Filter" on page 151. For more information about firewall filters, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

### Example: Applying the Inbound Traffic Filter to the Encryption Interface

Apply the inbound firewall filter (`ipsec-decrypt-policy-filter`) to the decrypted packet to perform the final policy check. The IPsec `manual-sa1` SA is referenced at the `[edit interfaces es-1/2/0 unit 0 family inet]` hierarchy level and decrypts the incoming packet.

The Packet Forwarding Engine directs IPsec packets to the ES PIC. It uses the packet's security parameter index (SPI), protocol, and destination address to look up the SA configured on one of the ES interfaces. The IPsec `manual-sa1` SA is referenced at the `[edit interfaces es-1/2/0 unit 0 family inet]` hierarchy level and is used to decrypt the incoming packet. When the packets are processed (decrypted, authenticated, or both), the input firewall filter (`ipsec-decrypt-policy-filter`) is applied on the decrypted packet to perform the final policy check. `term1` defines the decrypted (and verified) traffic and performs the required policy check. For information about `term1`, see "Example: Configuring an Inbound Traffic Filter" on page 151.

> **NOTE**: The inbound traffic filter is applied after the ES PIC has processed the packet, so the decrypted traffic is defined as any traffic that the remote gateway is encrypting and sending to this router. IKE uses this filter to determine the policy required for a tunnel. This policy is used during the negotiation with the remote gateway to find the matching SA configuration.

```
[edit interfaces]
es-1/2/0 {
    unit 0 {
        tunnel {
            source 10.5.5.5;                         # tunnel source address
            destination 10.6.6.6;                  # tunnel destination address
        }
        family inet {
            filter {
                input ipsec-decrypt-policy-filter;
            }
            ipsec-sa manual-sa1;                 # SA name applied to packet
            address 10.1.1.8/32 { # local interface address inside local VPN
            destination 10.2.2.254;          # destination address inside remote VPN
        }
```

```
    }
}
```

## Configuring an ES Tunnel Interface for a Layer 3 VPN

To configure an ES tunnel interface for a Layer 3 VPN, you need to configure an ES tunnel interface on the provider edge (PE) router and on the customer edge (CE) router. You also need to configure IPsec on the PE and CE routers. For more information about configuring an ES tunnel for a Layer 3 VPN, see the Junos OS VPNs Library for Routing Devices.

## Configuring ES PIC Redundancy

**IN THIS SECTION**



You can configure ES PIC redundancy on M Series and T Series routers that have multiple ES PICs. With ES PIC redundancy, one ES PIC is active and another ES PIC is on standby. When the primary ES PIC has a servicing failure, the backup becomes active, inherits all the tunnels and SAs, and acts as the new next hop for IPsec traffic. Reestablishment of tunnels on the backup ES PIC does not require new Internet Key Exchange (IKE) negotiations. If the primary ES PIC comes online, it remains in standby and does not preempt the backup. To determine which PIC is currently active, use the `show ipsec redundancy` command.

**NOTE**: ES PIC redundancy is supported on M Series and T Series routers.

To configure an ES PIC as the backup, include the `backup-interface` statement at the `[edit interfaces` *`fpc/pic/port`* `es-options]` hierarchy level:

```
backup-interface es-fpc/pic/port;
```

### Example: Configuring ES PIC Redundancy

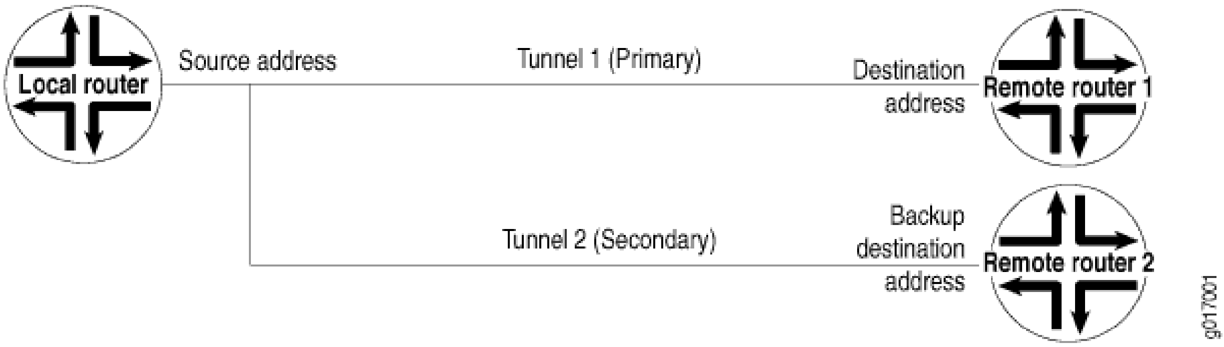
After you create the inbound firewall filter, apply it to the primary ES PIC. Here, the inbound firewall filter (`ipsec-decrypt-policy-filter`) is applied on the decrypted packet to perform the final policy check. The IPsec `manual-sa1` SA is referenced at the `[edit interfaces es-1/2/0 unit 0 family inet]` hierarchy level and decrypts the incoming packet. This example does not show SA and filter configuration. For information about SA and filter configuration, see the Junos OS Administration Library for Routing Devices, the Routing Policies, Firewall Filters, and Traffic Policers User Guide, and "Example: Configuring an Inbound Traffic Filter" on page 146.

```
[edit interfaces]
es-1/2/0 {
    es-options {
        backup-interface es-1/0/0;
    }
    unit 0 {
        tunnel {
            source 10.5.5.5;
            destination 10.6.6.6;
        }
        family inet {
            ipsec-sa manual-sa1;
            filter {
                input ipsec-decrypt-policy-filter;
            }
            address 10.1.1.8/32 {
                destination 10.2.2.254;
            }
        }
    }
}
```

## Configuring IPsec Tunnel Redundancy

You can configure IPsec tunnel redundancy by specifying a backup destination address. The local router sends keepalives to determine the remote site's reachability. When the peer is no longer reachable, a new tunnel is established. For up to 60 seconds during failover, traffic is dropped without notification being sent. Figure 8 on page 155 shows IPsec primary and backup tunnels.

**Figure 8: IPsec Tunnel Redundancy**




To configure IPsec tunnel redundancy, include the `backup-destination` statement at the `[edit interfaces unit` *`logical-unit-number`* `tunnel]` hierarchy level:

```
backup-destinationaddress;
destination address;
source address;
```

**NOTE**: Tunnel redundancy is supported on M Series and T Series routers.

The primary and backup destinations must be on different routers.

The tunnels must be distinct from each other and policies must match.

For more information about tunnels, see "Tunnel Interface Configuration on MX Series Routers Overview" on page 14.

3

CHAPTER

# Configuration Statements

# address (Interfaces)

IN THIS SECTION



## Syntax

```
address address {
    destination address;
}
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number family family]
```

## Description

Configure the interface address.

## Options

*address*—Address of the interface.

The remaining statement is explained separately. See CLI Explorer.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Encryption Interfaces | 144

# allow-fragmentation

IN THIS SECTION

## Syntax

```
allow-fragmentation;
```

## Hierarchy Level

```
[edit interfaces gr-fpc/pic/port unit logical-unit-number tunnel],
[edit logical-systems logical-system-name interfaces gr-fpc/pic/port unit logical-unit-number
tunnel]
```

## Description

For a generic routing encapsulation (GRE) tunnel, enable fragmentation of GRE-encapsulated packets whose size exceeds the maximum transmission unit (MTU) value of a link that the packet passes through. The don't-fragment (DF) bit is not set in the outer IP header of GRE-encapsulated packets.

To enable the reassembly of fragmented GRE-encapsulated packets on GRE tunnel interfaces at the endpoint of the GRE tunnel, include the "reassemble-packets" on page 219 statement for the interface.

> **NOTE**: The "reassemble-packets" on page 219 statement is not supported on MPC10E line card in Junos OS Release 19.3.

## Default

If you do not include the `allow-fragmentation` statement, fragmentation of GRE-encapsulated packets is disabled.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 9.2.

Statement introduced in Junos OS Release 19.3 for MPC10E line card.

RELATED DOCUMENTATION

reassemble-packets | 219
Filtering Unicast Packets Through Multicast Tunnel Interfaces | 73
Junos OS Services Interfaces Library for Routing Devices

# allow-fragmentation (fti)

IN THIS SECTION



## Syntax

```
allow-fragmentaion;
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number tunnel encapsulation (gre|udp|ipip)]
```

## Description

For an encapsulated (UDP/GRE/IP-in-IP) tunnel, enable fragmentation of GRE-encapsulated packets whose size exceeds the maximum transmission unit (MTU) value of a link that the packet passes through. The don’t-fragment (DF) bit is not set in the outer IP header of encapsulated packets.

## Default

Don't Frafgment (DF) bit is set by default for the outer encapsulated IPv4 header.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Evolved Release 23.2R1.

RELATED DOCUMENTATION

Tunnel Services Overview | 2

# apply-groups-except

IN THIS SECTION



## Syntax

```
apply-groups-except values;
```

## Hierarchy Level

```
[edit interfaces]
```

## Description

Disable inheritance of a configuration group.

## Options

*value*—.

## Required Privilege Level

configure—To enter configuration mode, but other required privilege levels depend on where the statement is located in the configuration hierarchy.

## Release Information

Statement introduced in Junos OS Release 15.1.

RELATED DOCUMENTATION

*groups*

*Disabling Inheritance of a Junos OS Configuration Group*

Tunnel Services Overview | 2

Tunnel Interface Configuration on MX Series Routers Overview | 14

# backup-destination

IN THIS SECTION

## Syntax

```
backup-destination destination-address;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number tunnel],[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number tunnel]
```

## Description

For tunnel interfaces, specify the remote address of the backup tunnel.

## Options

*destination-address*—Address of the remote side of the connection.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION


*destination (Interfaces)*

destination (Tunnel Remote End) | 182

Configuring IPsec Tunnel Redundancy | 154


# backup-interface

IN THIS SECTION



## Syntax

```
backup-interface interface-name;
```

## Hierarchy Level

```
[edit interfaces interface-name es-options]
```

## Description

Configure a backup ES Physical Interface Card (PIC). When the primary ES PIC has a servicing failure, the backup becomes active, inherits all the tunnels and security associations (SAs), and acts as the new next hop for IPsec traffic.

## Options

*interface-name*—Name of ES interface to serve as the backup.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring ES PIC Redundancy | 153

# bandwidth (Tunnel Services)

IN THIS SECTION

## Syntax

```
bandwidth bandwidth-value;
```

## Hierarchy Level

```
[edit chassis fpc slot-number pic number tunnel-services]
```

## Description

(ACX Series, MX Series 5G Universal Routing Platforms and T4000 Core Routers only) Configure the amount of bandwidth in gigabits per second reserved on each Packet Forwarding Engine for tunnel traffic using tunnel services. Configuring the bandwidth creates a virtual tunnel interface that is represented as *interface-<fpc/pic/port>*. Refer to Tunnel Services Overview for a list of interfaces that support tunneling services.

## Options

*bandwidth-value*—Amount of bandwidth in Gbps to reserve for tunnel traffic using tunnel services:

- On ACX Series routers, the bandwidth values can be `1g` or `10g`.
- On MX Series routers, the bandwidth values can be as follows:
  - `1g`

  - `10g` through `100g` in 10 Gbps increments: `10g`, `20g`, `30g`, `40g`, `50g`, `60g`, `70g`, `80g`, `90g`, `100g`
  - `100g` through `400g` in 100 Gbps increments: `100g`, `200g`, `300g`, `400g`
- On T4000 routers, the bandwidth values can be `10g` through `100g` in 10 Gbps increments: `10g`, `20g`, `30g`, `40g`, `50g`, `60g`, `70g`, `80g`, `90g`, `100g`.

**NOTE**: The bandwidth that you specify determines the port number of the tunnel interfaces that are created. When you specify a bandwidth of `1g`, the port number is always 10. When you specify any other bandwidth, the port number is always 0.

**NOTE**: If you specify a bandwidth that is not compatible with the type of DPCs or MPCs and their respective Packet Forwarding Engine, tunnel services are not activated. For example, you cannot specify 1 gigabit per second bandwidth for a Packet Forwarding Engine on a 10-Gigabit Ethernet 4-port DPC or 16x10GE 3D MPC.

When the tunnel bandwidth is unspecified in the Routing Engine CLI, the maximum tunnel bandwidth for MPC3E is 60G.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 8.2.

RELATED DOCUMENTATION

Example: Configuring Tunnel Interfaces on a Gigabit Ethernet 40-Port DPC

Tunnel Interface Configuration on MX Series Routers Overview

Configuring Tunnel Interfaces on T4000 Routers

Example: Configuring Tunnel Interfaces on a 10-Gigabit Ethernet 4-Port DPC

Example: Configuring Tunnel Interfaces on the MPC3E

*tunnel-services (Chassis)*

# clear-dont-fragment-bit (Interfaces GRE Tunnels)

IN THIS SECTION



## Syntax

```
clear-dont-fragment-bit;
```

## Hierarchy Level

```
[edit interfaces gr-fpc/pic/port unit logical-unit-number],
[edit logical-systems logical-system-name interfaces gr-fpc/pic/port unit logical-unit-number]
```

## Description

Clear the Don't Fragment (DF) bit on all IP version 4 (IPv4) packets entering the generic routing encapsulation (GRE) tunnel on Adaptive Services (AS) or Multiservices interfaces. If the encapsulated packet size exceeds the tunnel maximum transmission unit (MTU), the packet is fragmented before encapsulation. The statement is supported only on MX Series routers and all M Series routers except the M320 router.

When you configure the `clear-dont-fragment-bit` statement on an interface with the MPLS protocol family enabled, you must specify an MTU value. This MTU value must not be greater than maximum supported value, which is 9192.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

Statement introduced in Junos OS Release 10.2 for MPC1 and MPC1Q.

> **NOTE**: On SRX platforms the clearing of the DF bit on a GRE tunnel is supported only when the device is in packet or selective packet mode; This feature is not supported in flow mode. As a result, when in flow mode, a packet that exceeds the MTU of the GRE interface with the DF bit set is dropped, despite having the `clear-dont-fragment-bit` configured on the GRE interface.

RELATED DOCUMENTATION

Enabling Fragmentation on GRE Tunnels | 70

# clear-dont-fragment-bit (fti)

IN THIS SECTION



## Syntax

```
clear-dont-fragment-bit;
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number family inet address]
```

## Description

Clear the Don’t Fragment (DF) bit on all IP version 4 (IPv4) packets entering the (UDP/GRE/IP-in-IP) tunnel on FTI interfaces. If the encapsulated packet size exceeds the tunnel maximum transmission unit (MTU), the packet is fragmented before encapsulation.

When the option is not configured and the DF bit is set, the packets will be dropped if the packet size is larger than the MTU of the FTI.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Evolved Release 23.2R1.

RELATED DOCUMENTATION

Tunnel Services Overview | 2

# copy-tos-to-outer-ip-header

IN THIS SECTION



## Syntax

```
copy-tos-to-outer-ip-header;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number]
```

## Description

For GRE tunnel interfaces only, enable the inner IP header's ToS bits to be copied to the outer IP packet header.

To verify that this option is enabled at the interface level, use the `show interfaces` *`interface-name`* `detail` command.

## Default

If you omit this statement, the ToS bits in the outer IP header are set to 0.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 8.2.

RELATED DOCUMENTATION

Configuring Unicast Tunnels | 74

# core-facing

IN THIS SECTION



## Syntax

```
core-facing;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number family family],
```

## Description

Specifies that the VLAN is physically connected to a core-facing ISP router and ensures that the network does not improperly treat the interface as a client interface. When specified, the interface is inserted into the core-facing default mesh group where traffic from pseudowires that belong to the default mesh group is not forwarded on the core-facing link.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 15.1.

Support for EVPN-MPLS on PTX Series routers added in Junos OS Evolved Release 23.1R1.

RELATED DOCUMENTATION

Junos OS Subscriber Management and Services Library
Tunnel Services Overview | 2
Tunnel Interface Configuration on MX Series Routers Overview | 14

# destination (Interfaces)

IN THIS SECTION

## Syntax

```
destination address;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number tunnel],
[edit interfaces interface-name unit logical-unit-number family inet address address],
[edit interfaces interface-name unit logical-unit-number tunnel],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number
family inet address address]
```

## Description

For CoS on ATM interfaces, specify the remote address of the connection.

For point-to-point interfaces only, specify the address of the interface at the remote end of the connection.

For tunnel and encryption interfaces, specify the remote address of the tunnel.

## Options

*address*—Address of the remote side of the connection.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

*Configuring Linear RED Profiles on ATM Interfaces*

*Configuring Link and Multilink Services Logical Interfaces*

Configuring Encryption Interfaces | **144**

*Configuring Traffic Sampling on MX, M and T Series Routers*

*Configuring Flow Monitoring*

Configuring Unicast Tunnels | **74**

# destination-hostname

IN THIS SECTION



## Syntax

```
destination-hostname destination-hostname;
```

## Hierarchy Level

```
[edit interfaces name unit name tunnel]
```

## Description

Define the hostname of the tunnel destination for the IP tunnel interface.

After the DNS resolves the IP address for the hostname, the resolved address is used as tunnel destination address. The ipip interface (IFL) is not created unless the hostname is resolved.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 23.2R1 for SRX Series Firewalls

RELATED DOCUMENTATION

Tunnel Services Overview | 2

tunnel | 238

# destination (Routing Instance)

IN THIS SECTION



## Syntax

```
destination routing-instance-name;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number tunnel routing-instance]
```

## Description

Specify the destination routing instance that points to the routing table containing the tunnel destination address.

## Default

The default Internet routing table `inet.0`.

## Options

*`routing-instance-name`*—Name of the destination routing instance.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Tunnel Interfaces for Routing Table Lookup | 135

# destination (Tunnel Remote End)

IN THIS SECTION

## Syntax

```
destination destination-address;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number tunnel],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number
tunnel]
```

## Description

For tunnel interfaces, specify the remote address of the tunnel.

## Options

*destination-address*—Address of the remote side of the connection.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Unicast Tunnels | 74

*Configuring Traffic Sampling on MX, M and T Series Routers*

*Configuring Flow Monitoring*

# destination-networks

IN THIS SECTION



## Syntax

```
destination-networks prefix;
```

## Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name routing-options
dynamic-tunnels tunnel-name],
[edit logical-systems logical-system-name routing-options dynamic-tunnels tunnel-name rsvp-te
entry],
[edit logical-systems logical-system-name routing-options dynamic-tunnels tunnel-name],
[edit routing-instances routing-instance-name routing-options dynamic-tunnels tunnel-name],
[edit routing-instances routing-instance-name routing-options dynamic-tunnels tunnel-name  rsvp-
te entry],
[edit routing-options dynamic-tunnels tunnel-name],
[edit routing-options dynamic-tunnels tunnel-name rsvp-te entry]
```

## Description

Specify the IPv4 prefix range for the destination network. Only tunnels within the specified IPv4 prefix range can be created.

## Options

*prefix*—Destination prefix of the network.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

*Configuring GRE Tunnels for Layer 3 VPNs*

Dynamic Tunnels Overview | 13

Configuring RSVP Automatic Mesh

# destination-udp-port (FTI)

IN THIS SECTION



## Syntax

```
destination-udp-port destination-udp-port;
```

## Hierarchy Level

```
[edit interfaces name unit name tunnel encapsulation vxlan-gpe]
```

## Description

Assign a numeric value to write to the destination-udp-port-field to identify a Virtual Extensible LAN (VXLAN). The range is 1 through 65,535.

## Required Privilege Level

routing

## Release Information

Statement introduced in Junos OS Release 18.3R1.

RELATED DOCUMENTATION

Flexible Tunnel Interfaces Overview | 38
Configuring Flexible Tunnel Interfaces | 45
show interfaces fti | 358
vni(Interfaces) | 252
Example: Configuring Flexible Tunnel Interfaces on MX Series Routers | 50

# do-not-fragment

IN THIS SECTION

## Syntax

```
do-not-fragment;
```

## Hierarchy Level

```
[edit interfaces gr-fpc/pic/port unit logical-unit-number tunnel],
[edit logical-systems logical-system-name interfaces gr-fpc/pic/port unit logical-unit-number
tunnel]
```

## Description

For a generic routing encapsulation (GRE) tunnel, disable fragmentation of GRE-encapsulated packets. This sets the do-not-fragment (DF) bit in the outer IP header of the GRE-encapsulated packets so that they do not get fragmented anywhere in the path. When the size of a GRE-encapsulated packet is greater than the MTU of a link that the packet passes through, the GRE-encapsulated packet is dropped.

## Default

By default, fragmentation of GRE-encapsulated packets is disabled.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 9.2.

Statement introduced in Junos OS Release 19.3 for MPC10E line card.

RELATED DOCUMENTATION


allow-fragmentation | 160

reassemble-packets | 219

Filtering Unicast Packets Through Multicast Tunnel Interfaces | 73

Junos OS Services Interfaces Library for Routing Devices


# dynamic-tunnels

IN THIS SECTION



## Syntax

```
dynamic-tunnels tunnel-name {
    bgp-signal;
```

```
        destination-networks prefix;
        gre;
        ipip
        rsvp-te entry-name {
            destination-networks network-prefix;
            label-switched-path-template (Multicast) {
                default-template;
                template-name;
            }
        }
        source-address address;
        spring-te;
        traceoptions;
        tunnel-attributes name {
            dynamic-tunnel-anchor-pfe dynamic-tunnel-anchor-pfe;
            dynamic-tunnel-anti-spoof (off | on);
            dynamic-tunnel-gre-key
            dynamic-tunnel-mtu dynamic-tunnel-mtu;
            dynamic-tunnel-source-prefix dynamic-tunnel-source-prefix;
            dynamic-tunnel-type (BGP-SIGNAL | GRE | UDP | V4oV6);
        }
      v4ov6 ipv6-anycast-source-duplication;
  }
```

## Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name routing-
options],
[edit logical-systems logical-system-name routing-options],
[edit routing-instances routing-instance-name routing-options],
[edit routing-options]
```

## Description

Configure a dynamic tunnel between two PE routers.

**NOTE**: ACX Series routers do not support the `gre` statement.

Configure dynamic IPv4-over-IPv6 tunnels and define their attributes to forward IPv4 traffic over an IPv6-only network. IPv4 traffic is tunneled from customer premises equipment to IPv4-over-IPv6 gateways. You must also configure `extended-nexthop` option at `[edit protocols bgp family inet unicast]` hierarchy level to allow BGP to route IPv4 address families over an IPv6 session.

## Options

bgp-signal — Enable the creation of a tunnel signaled by BGP.

gre — Enable dynamic generic routing encapsulation type tunnel mode for IPv4.

- Values:
  - `next-hop-based-tunnel`—Enable next hop base dynamic-tunnel for steering IPv4 traffic with IPv6 next hop address.

ipip — Enable dynamic IP in IP encapsulation type tunnel mode for IPv4

- Values:
  - `full-resolved-next-hop-based-tunnel`—Enable fully resolved next hop base dynamic-tunnel for steering IPv4 traffic with IPv6 next hop address.

source-address — Specify the source address of the tunnel.

*`tunnel-name`* — Name of the dynamic tunnel.

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

`bgp-signal` option introduced in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

- extended-nexthop
- tunnel-attributes
- Example: Configuring a Two-Tiered Virtualized Data Center for Large Enterprise Networks
- Understanding Redistribution of IPv4 Routes with IPv6 Next Hop into BGP

# encap-type

IN THIS SECTION



## Syntax

```
encap-type(ipv4 | ipv6);
```

## Hierarchy Level

```
[edit interfaces name unit name tunnel]
```

## Description

Define the type of encapsulation on the IP tunnel interface .

You must configure this option if you configure `source-interface` or `source-hostname` or `destination-hostname`.

## Options

| | |
|---|---|
| ipv4 | For IPv4 tunnel encapsulation |
| ipv6 | For IPv6 tunnel encapsulation |

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 23.2R1 for SRX Series Firewalls

RELATED DOCUMENTATION

Tunnel Services Overview | 2

# es-options

IN THIS SECTION



## Syntax

```
es-options {
    backup-interface interface-name;
}
```

## Hierarchy Level

```
[edit interfaces interface-name]
```

## Description

On ES interfaces, configure ES interface-specific interface properties.

The `backup-interface` statement is explained separately.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring ES PIC Redundancy | 153

# fabric loopback wan

IN THIS SECTION



## Syntax

```
fabric loopback wan (off | on);
```

## Hierarchy Level

```
[edit chassis fpc slot-number]
```

## Description

Allow the fabric bound packets to take new fabric loopback path, freeing up the WAN bandwidth thus improving the sampling and tunneling performance of the router.

You can configure fabric side loopback by using the `fabric loopback wan off` option or switch to WAN side by using the `fabric loopback wan on` option. By default, Junos OS uses fabric loopback for the loopback packets.

## Default

`off`

## Options

`off`—Configure fabric bound packets to take fabric loopback path.

`on`—Configure fabric bound packets to take the WAN loopback path.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 20.3.

# family

IN THIS SECTION



## Syntax

```
family family {
    ipsec-sa sa-name;
}
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number]
```

## Description

Configure protocol family information for the logical interface.

## Options

*family*—Protocol family:

- ccc—Circuit cross-connect protocol suite
- inet—IP version 4 suite
- inet6—IP version 6 suite
- iso—Open Systems Interconnection (OSI) International Organization for Standardization (ISO) protocol suite
- mlfr-end-to-end—Multilink Frame Relay FRF.15
- mlfr-uni-nni—Multilink Frame Relay FRF.16
- multilink-ppp—Multilink Point-to-Point Protocol
- mpls—MPLS
- tcc—Translational cross-connect protocol suite
- tnp—Trivial Network Protocol
- vpls—Virtual private LAN service

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Encryption Interfaces | 144
Junos OS Network Interfaces Library for Routing Devices

# family bridge

IN THIS SECTION



## Syntax

```
family bridge {
    apply-groups value;
    apply-groups-except value;
    core-facing;
    interface-mode access | trunk;
    inner-vlan-id-list inner-vlan-id-range;
    storm-control;
    vlan-id vlan-id ;
```

```
    vlan-id-list vlan-id-range;
}
```

## Hierarchy Level

```
[edit interfaces]
[edit logical-systems name interfaces]
```

## Description

Family bridge is used when you want a port that has more than one logical unit, each with the same or different encapsulations. Bridge domains are associated with GRE interface with the corresponding BD VLAN.

## Options

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

system—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 15.1.

Starting in Junos OS release 17.4R1 for MX Series routers, support for storm control was added for logical systems.

RELATED DOCUMENTATION

Tunnel Services Overview | 2
Tunnel Interface Configuration on MX Series Routers Overview | 14

# family bridge (GRE Interfaces)

IN THIS SECTION



## Syntax

```
family bridge {
    interface-mode (access | trunk);
    core-facing;
    vlan-id number;
    vlan-id-list vlan-list;
}
```

## Hierarchy Level

```
[edit interfaces gr-fpc/pic/port.unit]
```

## Description

Configure the bridge domain family on GRE interfaces. To enable Layer 2 Ethernet packets to be terminated on GRE tunnels, you must configure the bridge domain protocol family on the `gr-` interfaces and associate the `gr-` interfaces with the bridge domain. You must configure the GRE interfaces as core-facing interfaces, and they must be access or trunk interfaces. To configure the bridge domain family on `gr-` interfaces, include the `family bridge` statement at the `[edit interfaces gr-`*`fpc/pic/port`*` unit `*`logical-unit-number`*`]` hierarchy level.

## Options

`interface-mode`—Specify the type of VLAN tagging on packets that the interface accepts.

`access`—Configure a logical interface to accept untagged packets. Specify the VLAN to which this interface belongs using the `vlan-id` statement.

`trunk`—Configure a single logical interface to accept packets tagged with any VLAN ID specified with the `vlan-id` or `vlan-id-list` statement.

`core-facing`—Specify that the VLAN is physically connected to a core-facing ISP router and ensures that the network does not improperly treat the interface as a client interface. When specified, the interface is inserted into the core-facing default mesh group where traffic from pseudowires that belong to the default mesh group is not forwarded on the core-facing link.

`vlan-id` *`number`*—Individual VLAN IDs separated by a space.

`vlan-id-list` *`vlan-list`*—Starting VLAN ID and ending VLAN ID in an inclusive range. Separate the starting VLAN ID and ending VLAN ID with a hyphen.

## Required Privilege Level

**interface**—To view this statement in the configuration.

**interface-control**—To add this statement to the configuration.

## Release Information

Statement at the `[edit interfaces gr-fpc/pic/port.unit]` hierarchy level introduced in Junos OS Release 15.1 for MX Series routers.

RELATED DOCUMENTATION

Layer 2 Services over GRE Tunnel Interfaces on MX Series with MPCs | 111

# filter

IN THIS SECTION



## Syntax

```
filter {
    input filter-name;
    output filter-name;
}
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number family inet]
```

## Description

Define the filters to be applied on an interface.

## Options

`input` *`filter-name`*—Identifier for the input filter.

`output` *`filter-name`*—Identifier for the output filter.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Filters for Traffic Transiting the ES PIC | 146

# hold-time (OAM)

IN THIS SECTION



## Syntax

```
hold-time seconds;
```

## Hierarchy Level

```
[edit protocols oam],
[edit protocols oam gre-tunnel interface interface-name]
```

## Description

Length of time the originating end of a GRE tunnel waits for keepalive packets from the other end of the tunnel before marking the tunnel as operationally down.

## Options

*seconds*—Hold-time value.

- **Default:** 5 seconds
- **Range:** 5 through 250 seconds

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 10.2.

RELATED DOCUMENTATION

Configuring Tunnel Interfaces on an MX Series Router with a 16x10GE 3D MPC | 16
Configuring GRE Keepalive Time | 66
keepalive-time | 211

# interfaces

IN THIS SECTION

## Syntax

```
interfaces { ... }
```

## Hierarchy Level

```
[edit]
```

## Description

Configure interfaces on the router.

## Default

The management and internal Ethernet interfaces are automatically configured. You must configure all other interfaces.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Junos OS Network Interfaces Library for Routing Devices

# ipip

IN THIS SECTION



## Syntax

```
ipip {
    destination {
        address address;
    }
    source {
        address address;
    }
}
```

## Hierarchy Level

```
[edit dynamic-profiles name interfaces name unit name tunnel encapsulation],
[edit dynamic-profiles name logical-systems name interfaces name unit name tunnel encapsulation],
[edit interfaces name unit name tunnel encapsulation]
```

## Description

Configure IP-in-IP tunnel.

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Release 20.3R1.

# ipsec-sa

IN THIS SECTION

## Syntax

```
ipsec-sa sa-name;
```

## Hierarchy Level

```
[edit interfaces es-fpc/pic/port unit logical-unit-number family inet]
```

## Description

Specify the IP Security (IPsec) SA name associated with the interface.

## Options

*sa-name*—IPsec SA name.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Encryption Interfaces | 144
Junos OS Administration Library for Routing Devices

# keepalive-time

IN THIS SECTION



## Syntax

```
keepalive-time seconds;
```

## Hierarchy Level

```
[edit protocols oam gre-tunnel interface interface-name],
[edit protocols oam gre-tunnel interface interface-name.unit-number]
```

## Description

Time difference between consecutive keepalive packets in a GRE tunnel.

> **NOTE**: Support for GRE keepalive packets on MPC line cards became available as of Junos OS Release 11.4.

## Options

*seconds*—Keepalive time value.

- **Default:** 1 second
- **Range:** 1 through 50 seconds

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 10.2.

Statement introduced in Junos OS Release 19.3 for MPC10E line card.

RELATED DOCUMENTATION



# key

IN THIS SECTION



## Syntax

```
key number;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number tunnel],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number
tunnel]
```

## Description

Identify an individual traffic flow within a tunnel, as defined in RFC 2890, *Key and Sequence Number Extensions to GRE*. On M Series and T Series routers, you can configure the GRE interface on an Adaptive Services, Multiservices, or Tunnel PIC. On MX Series routers, configure the interface on a Multiservices DPC.

## Options

*number*—Value of the key.

- **Range:** 0 through 4,294,967,295

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Unicast Tunnels | 74

# multicast-only

IN THIS SECTION



## Syntax

```
multicast-only;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number family inet],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number
family inet]
```

## Description

Configure the unit and family so that the interface can transmit and receive multicast traffic only. You can configure this property on the IP family only.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Restricting Tunnels to Multicast Traffic | 82

*tunnel*

# peer-unit

IN THIS SECTION



## Syntax

```
peer-unit unit-number;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number]
```

## Description

Configure a peer relationship between two logical systems.

## Options

*unit-number*—Peering logical system unit number.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Logical Tunnel Interfaces | 83

# peer-certificate-type

IN THIS SECTION



## Syntax

```
peer-certificate-type (pkcs7 | x509-signature);
```

## Hierarchy Level

```
[edit services ipsec-vpn ike policy policy-name]
```

## Description

(MX Series routers only) Specify a preferred type of certificate (PKCS7 or X509). By default, X509 encoding format is used. With the flexibility to configure the encoding format in which certificate requests are sent to the peer, you can determine the type of certificate to be used depending on the type supported by the peer. For example, if the peer does not support PKCS7, certificate authentication cannot occur unless you configure the same type on MX Series routers as the initiator or sender.

## Options

- `pkcs7`—Public-Key Cryptography Standard #7.
- `x509-signature`—X509 is an ITU-T standard for public key infrastructure.

## Required Privilege Level

system—To view this statement in the configuration.

system-control—To add this statement to the configuration.

## Release Information

Statement introduced in Release 15.1.

RELATED DOCUMENTATION

*Configuring IKE Policies*

# reassemble-packets

IN THIS SECTION

## Syntax

```
reassemble-packets;
```

## Hierarchy Level

```
[edit interfaces gr-fpc/pic/port unit logical-unit-number],
[edit logical-systems logical-system-name interfaces gr-fpc/pic/port unit logical-unit-number]
```

## Description

Enable reassembly of fragmented generic routing encapsulation (GRE) encapsulated packets on GRE tunnel interfaces at the endpoint of the GRE tunnel.

GRE-encapsulated packets are fragmented if the "allow-fragmentation" on page 160 statement is configured for the GRE tunnel and the size of the GRE-encapsulated packet exceeds the maximum transmission unit (MTU) value of a link that the packet passes through.

## Default

If you do not include the `reassemble-packets` statement, the GRE tunnel interface does not reassemble fragmented GRE-encapsulated packets.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 9.2.

RELATED DOCUMENTATION

Filtering Unicast Packets Through Multicast Tunnel Interfaces | 73

# redundancy-group (Interfaces)

IN THIS SECTION



## Syntax

```
redundancy-group {
    member-interface interface-name {
        (active | backup);
    }
}
```

## Hierarchy Level

```
[edit interfaces interface-name]
```

## Description

Configure member tunnels of redundant logical or virtual tunnels only on MX Series 5G Universal Routing Platforms.

## Options

| | |
|---|---|
| active | Set the interface to the active mode. |
| backup | Set the interface to the backup mode. |

The remaining statement is explained separately. See CLI Explorer.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To view this statement in the configuration.

## Release Information

Statement introduced in Junos OS Release 13.3.

RELATED DOCUMENTATION

*Redundant Virtual Tunnels Providing Resiliency in Delivering Multicast Traffic Overview*

*Configuring Redundant Virtual Tunnels to Provide Resiliency in Delivering Multicast Traffic*

# redundancy-group (Chassis - MX Series)

IN THIS SECTION



## Syntax

```
redundancy-group {
    interface-type {
        redundant-logical-tunnel {
            device count;
        }
        redundant-virtual-tunnel {
            device count;
        }
    }
}
```

## Hierarchy Level

```
[edit chassis]
```

## Description

Configure redundant logical tunnels, redundant virtual tunnels, or both on MX Series 5G Universal Routing Platforms.

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 13.3.

RELATED DOCUMENTATION

*Redundant Virtual Tunnels Providing Resiliency in Delivering Multicast Traffic Overview*

*Configuring Redundant Virtual Tunnels to Provide Resiliency in Delivering Multicast Traffic*

*Example: Configuring Redundant Virtual Tunnels to Provide Resiliency in Delivering Multicast Traffic*

redundancy-group (Interfaces) | 221

# routing-instance

IN THIS SECTION



## Syntax

```
routing-instance {
    destination routing-instance-name;
}
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number tunnel],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number
tunnel]
```

## Description

Specify the destination routing instance that points to the routing table containing the tunnel destination address.

## Default

The default Internet routing table `inet.0`.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Tunnel Interfaces for Routing Table Lookup | 135

# routing-instances

IN THIS SECTION

## Syntax

```
routing-instances routing-instance-name { ... }
```

## Hierarchy Level

```
[edit],
[edit logical-systems logical-system-name]
```

## Description

Configure an additional routing entity for a router or switch. You can create multiple instances of BGP, IS-IS, OSPF, OSPF version 3 (OSPFv3), and RIP for a router or switch.

## Default

Routing instances are disabled for the router or switch.

## Options

*routing-instance-name*—Name of the routing instance, a maximum of 31 characters. The remaining statements are explained separately.

*non-forwarding-vrf*—Enable this option to not create a routing and forwarding (VRF) table for local or transit routes belonging to the given VPN.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

*Configuring EVPN Routing Instances*

*Configuring Routing Instances on PE Routers in VPNs*

# routing-options

IN THIS SECTION



## Syntax

```
routing-options { ... }
```

For information on the complete list of `routing-options`, see the Protocol-Independent Routing Properties User Guide .

## Hierarchy Level

```
[edit],
[edit logical-systems logical-system-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit tenants tenant-name routing-instances routing-instance-name],
[edit routing-instances routing-instance-name]
```

## Description

Configure protocol-independent routing properties.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

The [edit tenants *tenant-name* routing-instances *routing-instance-name*] hierarchy level introduced in Junos OS Release 18.3R1.

RELATED DOCUMENTATION

Protocol-Independent Routing Properties User Guide

# source

IN THIS SECTION



## Syntax

```
source source-address;
```

## Hierarchy Level (EX, NFX, OCX1100 and QFX Series)

```
[edit interfaces interface-name unit logical-unit-number tunnel]
```

## Hierarchy Level (M-series and T-series)

```
[edit interfaces interface-name unit logical-unit-number tunnel address],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number
tunnel address]
```

## Description

Specify the source address of the tunnel.

## Default

If you do not specify a source address, the tunnel uses the unit's primary address as the source address of the tunnel.

## Options

*source-address*—Address of the local side of the tunnel. This is the address that is placed in the outer IP header's source field.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Tunnel Services Overview | 2
multicast-only
*primary (Address on Interface)*
Junos OS Services Interfaces Library for Routing Devices

# source-address

IN THIS SECTION



## Syntax

```
source-address address;
```

## Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name routing-
options dynamic-tunnels tunnel-name],
[edit logical-systems logical-system-name routing-options dynamic-tunnels tunnel-name],
[edit routing-instances routing-instance-name routing-options dynamic-tunnels tunnel-name],
[edit routing-options dynamic-tunnels tunnel-name]
```

## Description

Configure the tunnel source address.

## Options

*address*—Name of the source address.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Dynamic Tunnels Overview | 13

# source-hostname

IN THIS SECTION

## Syntax

```
source-hostname source-hostname;
```

## Hierarchy Level

```
[edit interfaces name unit name tunnel]
```

## Description

Define the hostname of the tunnel source for the IP tunnel interface.

After the DNS resolves the IP address for the hostname, the resolved address is used as tunnel source address. The ipip interface (IFL) is not created unless the hostname is resolved.

.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 23.2R1 for SRX Series Firewalls

RELATED DOCUMENTATION


Tunnel Services Overview | 2

tunnel | 238


# source-interface

IN THIS SECTION



## Syntax

```
source-interface source-interface;
```

## Hierarchy Level

```
[edit interfaces name unit name tunnel]
```

## Description

Define the tunnel source interface. IP address from source-interface is used as tunnel source address.

When you configure IP adddress for the tunnel source-interface using a CLI, the IP tunnel derives that IP address and uses it as tunnel source address.

When you configure the tunnel source-interface to derive IP address from the DHCP server, IP tunnel derives the address once the source interface gets the address from the DHCP server.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 23.2R1 for SRX Series Firewalls

RELATED DOCUMENTATION

Tunnel Services Overview | 2
tunnel | 238

# ttl

IN THIS SECTION

## Syntax

```
ttl value;
```

## Hierarchy Level

```
[edit interfaces interface-name unit number tunnel]
```

## Description

Set the time-to-live value bit in the header of the outer IP packet.

## Options

*value*—Time-to-live value.

- **Range:** 0 through 255
- **Default:** 64

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Tunnel Services Overview | 2

# tunnel

IN THIS SECTION



## Syntax

```
tunnel {
    backup-destination destination-address;
    destination destination-address;
    encapsulation {
        gre
            destination {
                addressaddress;
```

```
            }
            key key;
            source {
                addressaddress;
            }
            tunnel-routing-instance {
                routing-instance routing-instance;
            }

        }
        ipip {
            destination {
                address address;
            }
            source {
                address address;
                interface interface;
            }
        }
    }
    routing-instance {
        destination routing-instance-name;
    }
    source source-address;
    ttl number;
}
```

## Syntax (SRX Series)

```
tunnel {
     destination;
destination-hostname;
    do-not-fragment;
    encap-type;
    flow-label;
    no-path-mtu-discovery;
    path-mtu-discovery;
 routing-instance {
        destination routing-instance-name;
```

```
    }
            source;
source-hostname;
          source-interface ;
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number]
```

## Description

Configure a tunnel. You can use the tunnel for unicast and multicast traffic or just for multicast traffic. You can also use tunnels for encrypted traffic or virtual private networks (VPNs).

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Encryption Interfaces | 144

Tunnel Services Overview | 2

# tunnel

IN THIS SECTION



## Syntax

```
tunnel {
    allow-fragmentation;
    backup-destination address;
    destination destination-address;
    do-not-fragment;
    key number;
    routing-instance {
        destination routing-instance-name;
    }
    source source-address;
    traffic-class traffic-class-value;
    ttl number;
}
```

## Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number],
[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number]
```

## Description

Configure a tunnel. You can use the tunnel for unicast and multicast traffic or just for multicast traffic. You can also use tunnels for encrypted traffic or virtual private networks (VPNs).

The remaining statements are explained separately. See CLI Explorer.

## Options

`encapsulation ipip ( source | address)` —(Optional) Configure IP-IP encapsulation type. To configure an IP-in-IP tunnel at an FTI, use the `encapsulation ipip` option at the `[edit interfaces interface-name unit logical-unit-number tunnel]` hierarchy level. The `source` and `destination` options support IPv4 addresses. You can also specify interface name instead of source address.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Encryption Interfaces | 144

# tunnel-port

IN THIS SECTION



## Syntax

```
tunnel-port port-number
```

## Hierarchy Level

```
[edit chassis fpc slot-number pic number tunnel-services]
```

## Description

For MX Series 5G Universal Routing Platforms, configure the port number of the tunnel. The range is 0 through 3.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 21.4R1.

RELATED DOCUMENTATION

*tunnel-services*

*bandwidth*

# tunnel-services (Chassis)

IN THIS SECTION



## Syntax

```
tunnel-services {
    bandwidth bandwidth-value;
    tunnel-only;
}
```

## Hierarchy Level

```
[edit chassis fpc slot-number pic number]
```

## Description

For MX Series 5G Universal Routing Platforms, configure the amount of bandwidth for tunnel services.

For ACX Series routers, configure the amount of bandwidth for tunnel services. Only bandwidths of 1 Gbps and 10 Gbps are supported for ACX routers.

For M7i, M10i, M120, M320, T Series and TX Matrix routers with IQ2 PICs and IQ2E PICs, configure support for per unit scheduling for GRE tunnels. You can specify the IQ2 and IQ2E PICs to work exclusively in tunnel mode or as a regular PIC. The default setting uses IQ2 and IQ2E PICs as a regular PIC. If you do not configure the `tunnel-only` option, the IQ2 and IQ2 PICs operate as regular PICs. For M7i, M10i, M120, M320, T Series and TX Matrix routers with IQ2 PICs and IQ2E PICs, you can use the `tunnel-only` option to specify that an IQ2 or IQ2E PIC work in tunnel mode only.

> **NOTE**: Bandwidth rates of 20 gigabits per second and 40 gigabits per second require use of an MX Series router with the 100-Gigabit Ethernet Modular Port Concentrator (MPC) and the 100-Gigabit CFP MIC.

> **NOTE**: On MX80 routers and MX Series routers with Trio-based FPCs, when ingress queuing is enabled for a PIC, tunnel services and inline services are not supported on the same PIC.

## Options

`tunnel-only` (Optional)—For M7i, M10i, M120, M320, T Series and TX Matrix routers with IQ2 PICs and IQ2E PICs, specify that an IQ2 or IQ2E PIC work in tunnel mode only.

The remaining statements are explained separately. Search for a statement in CLI Explorer or click a linked statement in the Syntax section for details.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 8.2.

RELATED DOCUMENTATION

Example: Configuring Tunnel Interfaces on a Gigabit Ethernet 40-Port DPC

Example: Configuring Tunnel Interfaces on a 10-Gigabit Ethernet 4-Port DPC

Example: Configuring Tunnel Interfaces on the MPC3E

*bandwidth (Tunnel Services)*

# udp (FTI)

IN THIS SECTION

## Syntax

```
udp {
    destination {
        address ipv4_address;
    }
    source {
        address ipv4_address;
        interface interface;
    }
}
```

## Hierarchy Level

```
[edit interfaces name unit name tunnel encapsulation]
```

## Description

Enable UDP encapsulation with IPv4 on FTI logical interfaces.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Release 19.3R1.

RELATED DOCUMENTATION

Flexible Tunnel Interfaces Overview | 38

# unit (Interfaces)

IN THIS SECTION



## Syntax

```
unit logical-unit-number {
    family inet {
        ipsec-sa sa-name;
    }
    tunnel {
        backup-destination destination-address;
        destination destination-address;
        routing-instance {
            destination routing-instance-name;
        }
        source source-address;
        ttl number;
    }
}
```

## Hierarchy Level

```
[edit interfaces interface-name]
```

## Description

Configure a logical interface on the physical device. You must configure a logical interface to be able to use the physical device.

## Options

*logical-unit-number*—Number of the logical unit.

- **Range:** 0 through 16,384

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Configuring Encryption Interfaces | 144
Junos OS Network Interfaces Library for Routing Devices
Junos OS Network Interfaces Library for Routing Devices

# unit (Interfaces)

IN THIS SECTION



## Syntax

```
unit logical-unit-number {
    peer-unit unit-number;
    reassemble-packets;
    tunnel {
        allow-fragmentation;
        backup-destination address;
        destination destination-address;
        do-not-fragment;
        key number;
        routing-instance {
            destination routing-instance-name;
        }
        source source-address;
        ttl number;
    }
}
```

## Hierarchy Level

```
[edit interfaces interface-name],
[edit logical-systems logical-system-name interfaces interface-name]
```

## Description

Configure a logical interface on the physical device. You must configure a logical interface to be able to use the physical device.

## Options

*logical-unit-number*—Number of the logical unit.

- **Range:** 0 through 16,384

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

Junos OS Network Interfaces Library for Routing Devices

# vni(Interfaces)

IN THIS SECTION



## Syntax

```
vni [0-16777214]
```

## Hierarchy Level

```
[edit interfaces name unit name tunnel encapsulation vxlan-gpe]
```

## Description

Assign a numeric value to identify a Virtual Extensible LAN (VXLAN). All members of a VXLAN must use the same virtual network identifier (VNI).

## Options

**vni** Value to specify in the `vni` attribute.

- **Range:** For flexible tunnel interfaces, the range supported is 0 through 16,777,214 for all MX Series routers (except MX80).

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Release 18.3R1.

RELATED DOCUMENTATION



# vxlan-gpe (FTI)

**IN THIS SECTION**

## Syntax

```
vxlan-gpe {
    destination {
        address address;
    }
    destination-udp-port destination-udp-port;
    source {
        address address;
        interface interface;
    }
    source-udp-port-range {
        (min min-port-number| max max-port-number)
    }
tunnel-endpoint vxlan;
    vni vni;
}
```

## Hierarchy Level

```
[edit dynamic-profiles name interfaces name unit name tunnel encapsulation],
[edit interfaces name unit name tunnel encapsulation]
```

## Description

Virtual Extensible LAN (VXLAN) is defined as an encapsulation format that encapsulates Ethernet frames in an outer UDP/IP transport. The `vxlan-gpe` encapsulation refers to the header `gpe` added to the encapsulation `vxlan` to extend the existing VXLAN protocol to provide support for multiprotocol encapsulation.

## Options

**destination-udp-port**

Value to write to the destination-udp-port field.

- **Range:** 1 through 65,535

**source-udp-port-range**

Value to write to the source-udp-port-range field.

- **min** ***min-port-number***: 1 through 65,535.
- **max** ***max-port-number***: 1 through 65,535

*min-port-number* should be less than or equal to *max-port-number*

If you do not specify the source port range, it is randomly set (in the range 49152 – 65535) based on the hash value calculated using various packet headers field.

**tunnel-endpoint**

Tunnel endpoint type.

- Values:
    - vxlan—To configure VXLAN tunnel endpoint.

**vni**

Value to specify in the `vni` attribute.

- **Range:** 0 through 16,777,214

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Release 18.3R1.

RELATED DOCUMENTATION

4

CHAPTER

# Configuration Statements: Generic Routing Encapsulation (GRE) Tunnel, IPIP, and UDP Encapsulation and Decapsulation using Flexible Tunnel Interfaces (FTIs)

# address

IN THIS SECTION



## Syntax

```
address address;
```

## Hierarchy Level

```
[edit interfaces name unit name tunnel encapsulation( gre | ipip | udp ) destination]
```

## Description

Interface address prefix for tunnel destination.

## Required Privilege Level

interface

## Release Information

Statement introduced for GRE tunnels in Junos OS Evolved Release 20.2R1.

Statement introduced for IPIP tunnels in Junos OS Evolved Release 20.4R1.

Statement introduced for UDP tunnels in Junos OS Evolved Release 21.1R1.

# destination

IN THIS SECTION



## Syntax

```
destination {
    address address;
}
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number tunnel encapsulation ( gre | ipip | udp )]
```

## Description

Interface address prefix of the destination where the tunnel terminates. Configuring this interface is mandatory.

> **NOTE**: Starting in Junos OS Evolved Release 20.3R1, for GRE tunnels, configure the address for the tunnel encapsulation destination address.
>
> When you do not configure "tunnel-termination" on page 274, you can use `destination address` *address* to configure the destination IP address in the outer IP header while encapsulating the packet.

## Options

**address** You can configure the destination address with an optional destination mask. The mask range is from 0 to 32 for IPv4, and 0 to 128 for IPv6 addresses. /32 is the default mask for IPv4, and /128 is the default mask for IPv6.

> **NOTE**: Starting in Junos OS Evolved Release 20.3R1, for GRE tunnels, if you do not configure "tunnel-termination" on page 274 for tunnel decapsulation, you cannot configure the mask for the destination address.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Evolved Release 20.2R1.

Statement introduced for IPIP tunnels in Junos OS Evolved Release 20.4R1.

Statement introduced for UDP tunnels in Junos OS Evolved Release 21.1R1.

# encapsulation

IN THIS SECTION



## Syntax

```
encapsulation {
    ( gre | ipip | udp )
        destination {
            addressaddress;
        }
        key key;
        source {
            addressaddress;
        }
        tunnel-routing-instance {
            routing-instance routing-instance;
            port-profile profile-name
        }
        tunnel-termination;
    }
     {
        destination {
            address address;
        }
        source {
            address address;
            interface interface;
        }
```

```
    }
}
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number tunnel]
```

## Description

Configure encapsulation over tunnel.

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Evolved Release 20.2R1.

Statement introduced for IPIP tunnels in Junos OS Evolved Release 20.4R1.

Statement introduced for UDP tunnels in Junos OS Evolved Release 21.1R1.

# gre

IN THIS SECTION



## Syntax

```
gre {
    destination {
        addressaddress;
    }
    key key; key;
    source {
        addressaddress;
    }
    tunnel-routing-instance {
        routing-instance routing-instance;
    }
    tunnel-termination;
}
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number tunnel encapsulation]
```

## Description

Configure GRE (Generic Routing Encapsulation) encapsulation type.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Evolved Release 20.2R1.

# ipip

IN THIS SECTION



## Syntax

```
ipip {
    destination {
        address address;
    }
    source {
```

```
        address address;
    }
}
```

## Hierarchy Level

```
[edit dynamic-profiles name interfaces name unit name tunnel encapsulation],
[edit dynamic-profiles name logical-systems name interfaces name unit name tunnel encapsulation],
[edit interfaces name unit name tunnel encapsulation]
```

## Description

Configure IP-in-IP tunnel.

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Release 20.3R1.

# encapsulation-udp-fti

IN THIS SECTION



## Syntax

```
encapsulation {
    udp
        destination {
            addressaddress;
        }
        key key;
        source {
            addressaddress;
        }
        tunnel-routing-instance {
            routing-instance routing-instance;
        }
        tunnel-termination;
port-profile profile-name;

    }

    }
}
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number tunnel]
```

## Description

Configure UDP encapsulation over tunnel.

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Evolved Release 21.1R1.

# key

IN THIS SECTION

## Syntax

```
key number;
```

## Hierarchy Level

```
[edit interfaces ftinumber unit logical-unit-number tunnelencapsulation ( gre | ipip | udp )]
```

## Description

Key for configuring the tunnel. Identifies an individual traffic flow within a tunnel.

## Options

number An identifier for a individual traffic flow within a GRE tunnel.

- **Range:** 1 through 65535.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Evolved Release 20.2R1.

Statement introduced for IPIP tunnels in Junos OS Evolved Release 20.4R1.

Statement introduced for UDP tunnels in Junos OS Evolved Release 21.1R1.

# mpls(forwarding-options)

IN THIS SECTION



## Syntax

```
mpls port-number
```

## Hierarchy Level

```
[edit security forwarding-options tunnel udp payload-port-profile profile-name]
```

## Description

Enable the forwarding of MPLS traffic.

## Default

If port number is not configured for MPLS, the default value of the port number is 6635.

## Required Privilege Level

security—To view this statement in the configuration.

security-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos Evolved Release 21.4R1 for the PTX Series routers.

RELATED DOCUMENTATION

MPLS Overview

# source

IN THIS SECTION

## Syntax

```
source {
    address address;
}
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number tunnel encapsulation ( gre | ipip | udp )]
```

## Description

Specify the source address of the tunnel.

## Default

Source address is not required. Not configuring a source address indicates that the tunnel will terminate on the destination address.

**NOTE**: Starting in Junos OS Evolved Release 20.3R1, when you do not configure the "tunnel-termination" on page 274, configuring the source address for the tunnel encapsulation is mandatory.

## Options

| | |
|---|---|
| address | Interface address prefix of the source address. |

## Required Privilege Level

interface

## Release Information

Statement introduced for GRE tunnels in Junos OS Evolved Release 20.2R1.

Statement introduced for IPIP tunnels in Junos OS Evolved Release 20.4R1.

Statement introduced for UDP tunnels in Junos OS Evolved Release 21.1R1.

# tunnel (fti)

IN THIS SECTION



## Syntax

```
tunnel {
    encapsulation {
        ( gre  | ipip | udp )
            destination {
                addressaddress;
            }
            key  key;  key;
            source {
```

```
                address address;
            }
            tunnel-routing-instance {
                routing-instance routing-instance;
            }
            tunnel-termination;
        }
    }
}
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number],
```

## Description

Configure a tunnel on an FTI interface to perform tunnel decapsulation.

The remaining statements are explained separately. See CLI Explorer.

## Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

## Release Information

Statement introduced in Junos OS Evolved Release 20.2R1.

Statement introduced for IPIP tunnels in Junos OS Evolved Release 20.4R1.

Statement introduced for UDP tunnels in Junos OS Evolved Release 21.1R1.

# tunnel-termination

IN THIS SECTION



## Syntax

```
tunnel-termination;
```

## Hierarchy Level

```
[edit interfaces fti number unit logical-unit-number tunnel encapsulation  ( gre | ipip | udp )]
[edit interfaces interface-name family inet]
[edit forwarding-options]
```

## Description

Enables the tunnel to terminate on the tunnel underlay interface configured at [edit interfaces *interface-namefamily* inet] hierarchy level.

On a flexible tunnel interface (FTI), enables tunnel termination for GRE, IPIP , or UDP decapsulation.

**NOTE**: Starting in Junos OS Evolved Release 20.3R1, this statement is optional. If the statement is configured, the router can perform decapsulation of packets only. When the statement is not configured, you can configure encapsulation and decapsulation on the FTI.

**NOTE**: For the Junos OS Evolved Release 20.1R2, you have to configure the `tunnel-termination` option for IP-IP decapsulation mandatorily. In Junos OS Evolved Release 20.4R1, when tunnel-termination is configured, the router functions in decap-only mode and the understanding of source address and destination address is swapped.

When configured at the [edit forwarding-options] hierarchy level, the `tunnel-termination` option enables the termination of tunnels on all interfaces at the global level. This option is supported only for EVPN-VXLAN and is disabled by default. It is mandatory to enable this option for EVPN-VXLAN deployments.

## Required Privilege Level

interface

## Release Information

Statement introduced in Junos OS Evolved Release 20.2R1.

Statement introduced for IPIP tunnels in Junos OS Evolved Release 20.4R1.

Statement introduced for UDP tunnels in Junos OS Evolved Release 21.1R1.

Statement introduced at the [edit forwarding-options] hierarchy level in Junos OS Evolved Release 21.4R2.

5

CHAPTER

# Operational Commands

# clear ike security-associations

IN THIS SECTION



## Syntax

```
clear ike security-associations
<destination-ip-address>
```

## Description

(Encryption interface on M Series and T Series routers only) Clear information about the current Internet Key Exchange (IKE) security association. This command is valid for dynamic security associations only. For IKEv2, this command creates new security associations for IKE SA and IPSEC SAs.

## Options

| | |
|---|---|
| none | Clear all IKE security associations. |
| *destination-ip-address* | (Optional) Clear the IKE security association at the specified destination address. |

## Required Privilege Level

view

## Output Fields

When you enter this command, you are provided feedback on the status of your request.

## Sample Output

### clear ike security-associations

```
user@host> clear ike security-associations
```

## Release Information

Command introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

*show ike security-associations*

# clear ipsec security-associations

IN THIS SECTION

## Syntax

```
clear ipsec security-associations
<sa-name>
```

## Description

(Encryption interface on M Series and T Series routers only) Clear information about the current IP Security (IPsec) security association. This command is valid for dynamic security associations only. For IKEv1, this command creates new security associations for IKE SA and IPSEC SAs.

## Options

| | |
|---|---|
| none | Clear all IPsec security associations. |
| *sa-name* | (Optional) Clear the specified security association. |

## Required Privilege Level

view

## Output Fields

See the *show ipsec security-associations* for an explanation of output fields.

## Sample Output

### clear ipsec security-associations

The following output from the `show ipsec security-associations detail` command is displayed before and after the `clear ipsec security-associations` command is issued:

```
user@host> show ipsec security-associations detail
Security association: sa-dynamic, Interface family: Up

  Direction: inbound, SPI: 242379418, State: Installed
  Mode: tunnel, Type: dynamic
  Protocol: ESP, Authentication: hmac-md5-96, Encryption: None
  Soft lifetime: Expires in 22979 seconds
  Hard lifetime: Expires in 28739 seconds

  Direction: outbound, SPI: 368592771, State: Installed
  Mode: tunnel, Type: dynamic
  Protocol: ESP, Authentication: hmac-md5-96, Encryption: None
  Soft lifetime: Expires in 22979 seconds
  Hard lifetime: Expires in 28739 seconds


user@host> clear ipsec security-associations


user@host> show ipsec security-associations detail
Security association: sa-dynamic, Interface family: Up

  Direction: inbound, SPI: 1031597683, State: Installed
  Mode: tunnel, Type: dynamic
  Protocol: ESP, Authentication: hmac-md5-96, Encryption: None
  Soft lifetime: Expires in 23037 seconds
  Hard lifetime: Expires in 28797 seconds
```

```
  Direction: outbound, SPI: 1618419878, State: Installed
  Mode: tunnel, Type: dynamic
  Protocol: ESP, Authentication: hmac-md5-96, Encryption: None
  Soft lifetime: Expires in 23037 seconds
  Hard lifetime: Expires in 28797 seconds
```

## Release Information

Command introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

*show ipsec security-associations*

# request ipsec switch

IN THIS SECTION



## Syntax

```
request ipsec switch (interface <es-fpc/pic/port> | security-associations <sa-name>)
```

## Description

(Encryption interface on M Series, PTX Series, and T Series routers and EX Series switches only) Manually switch from the primary to the backup encryption services interface, or switch from the primary to the backup IP Security (IPsec) tunnel.

## Options

| | |
|---|---|
| interface <es-*fpc*/*pic*/*port*> | Switch to the backup encryption interface. |
| security-associations <*sa-name*> | Switch to the backup tunnel. |

## Required Privilege Level

view

## Output Fields

When you enter this command, you are provided feedback on the status of your request.

## Sample Output

### request ipsec switch security-associations

```
user@host> request ipsec switch security-associations sa-private
```

## Release Information

Command introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

show ipsec redundancy | 377

# request security certificate enroll (Signed)

IN THIS SECTION



## Syntax

```
request security certificate enroll filename filename  subject subject
alternative-subject alternative-subject  certification-authority certification-authority
encoding (binary | pem) key-file key-file  domain-name domain-name
```

## Description

(Encryption interface on M Series and T Series routers and EX Series switches only) Obtain a signed certificate from a certificate authority (CA). The signed certificate validates the CA and the owner of the certificate. The results are saved in a specified file to the **/var/etc/ikecert** directory.

**NOTE**: For FIPS mode, the digital security certificates must be compliant with the National Institute of Standards and Technology (NIST) SP 800-131A standard. The `request security key-pair` command is deprecated and not available with Junos in FIPS mode because it generates RSA and DSA keys with sizes of 512 and 1024 bits that are not compliant with the NIST SP 800-131A standard.

## Options

| | |
|---|---|
| filename *filename* | File that stores the certificate. |
| subject *subject* | Distinguished name (**dn**), which consists of a set of components—for example, an organization (**o**), an organization unit (**ou**), a country (**c**), and a locality (**l**). |
| alternative-subject *alternative-subject* | Tunnel source address. |
| certification-authority *certification-authority* | Name of the certificate authority profile in the configuration. |
| encoding (binary \| pem) | File format used for the certificate. The format can be a binary file or privacy-enhanced mail (PEM), an ASCII base64-encoded format. The default format is binary. |
| key-file *key-file* | File containing a local private key. |
| domain-name *domain-name* | Fully qualified domain name. |

## Required Privilege Level

maintenance

## Output Fields

When you enter this command, you are provided feedback on the status of your request.

## Sample Output

**request security certificate enroll filename subject alternative-subject certification-authority key-file domain-name (Signed)**

```
user@host> request security certificate enroll filename host.crt subject c=uk,o=london
alternative-subject 10.50.1.4 certification-authority verisign                 key-file
host-1.prv domain-name host.example.com
CA name: example.com CA file: ca_verisign
local pub/private key pair: host.prv
subject: c=uk,o=london domain name: host.example.com
alternative subject: 10.50.1.4
Encoding: binary
Certificate enrollment has started. To view the status of your enrollment, check the key
management process (kmd) log file at /var/log/kmd. <--------------
```

## Release Information

Command introduced before Junos OS Release 7.4.

# request security certificate enroll (Unsigned)

IN THIS SECTION

## Syntax

```
request security certificate enroll filename filename  ca-file ca-file  ca-name ca-
name
encoding (binary | perm) url url
```

## Description

(Encryption interface on M Series and T Series routers and EX Series switches only) Obtain a certificate from a certificate authority (CA). The results are saved in a specified file to the **/var/etc/ikecert** directory.

## Options

| | |
|---|---|
| filename *filename* | File that stores the public key certificate. |
| ca-file *ca-file* | Name of the certificate authority profile in the configuration. |
| ca-name *ca-name* | Name of the certificate authority. |
| encoding (binary \| pem) | File format used for the certificate. The format can be a binary file or privacy-enhanced mail (PEM), an ASCII base64-encoded format. The default value is **binary**. |
| url *url* | Certificate authority URL. |

## Required Privilege Level

maintenance

## Output Fields

When you enter this command, you are provided feedback on the status of your request.

## Sample Output

### request security certificate enroll filename ca-file ca-name url (Unsigned)

```
user@host> request security certificate enroll filename ca_verisign ca-file verisign ca-name
example.com urlxyzcompany URL
http://<verisign ca-name xyzcompany url>/cgi-bin/pkiclient.exe CA name: example.com CA file:
verisign Encoding: binary
Certificate enrollment has started. To view the status of your enrollment, check the key
management process (kmd) log file at /var/log/kmd. <--------------
```

## Release Information

Command introduced before Junos OS Release 7.4.

# request security key-pair

IN THIS SECTION

## Syntax

```
request security key-pair filename
<size key-size>
<type (rsa | dsa)>
```

## Description

(Encryption interface on M Series and T Series routers and EX Series switches only) Generate a public and private key pair for a digital certificate.

> **NOTE**: The `request security-certificates` command is deprecated and are not available with Junos in FIPS mode because security certificates are not compliant with the NIST SP 800-131A standard.

## Options

*filename* — Name of a file in which to store the key pair.

size *key-size* — (Optional) Key size, in bits. The key size can be **512**, **1024**, or **2048**. The default value is 1024.

type — (Optional) Algorithm used to encrypt the key:

- **rsa**—RSA algorithm. This is the default.
- **dsa**—Digital signature algorithm with Secure Hash Algorithm (SHA).

## Required Privilege Level

maintenance

## Output Fields

When you enter this command, you are provided feedback on the status of your request.

## Sample Output

### request security key-pair

```
user@host> request security key-pair security-key-file
```

## Release Information

Command introduced before Junos OS Release 7.4.

# request system certificate add

IN THIS SECTION

## Syntax

```
request system certificate add (filename | terminal)
```

## Description

(Encryption interface on M Series and T Series routers, PTX Series, and QFX Series switches only) Add a certificate provided by the Juniper Networks certificate authority (CA).

## Options

| | |
|---|---|
| *filename* | Filename (URL, local, or remote). |
| terminal | Use login terminal. |

## Required Privilege Level

maintenance

## Output Fields

When you enter this command, you are provided feedback on the status of your request.

## Sample Output

### request system certificate add terminal

```
user@host> request system certificate add terminal
```

## Release Information

Command introduced before Junos OS Release 7.4.

# show ike security-associations

IN THIS SECTION



## Syntax

```
show ike security-associations
<brief | detail>
<peer-address>
```

## Description

(Encryption interface on M Series and T Series routers only) Display information about Internet Key Exchange (IKE) security associations.

## Options

| | |
|---|---|
| none | Display standard information about all IKE security associations. |
| brief \| detail | (Optional) Display the specified level of output. |
| *peer-address* | (Optional) Display IKE security associations for the specified peer address. |

## Required Privilege Level

view

## Output Fields

Table 11 on page 293 lists the output fields for the `show ike security-associations` command. Output fields are listed in the approximate order in which they appear.

**Table 11: show ike security-associations Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **IKE peer** | Remote end of the IKE negotiation. | **detail** |
| **Role** | Part played in the IKE session. The router triggering the IKE negotiation is the initiator, and the router accepting the first IKE exchange packets is the responder. | **detail** |
| **Remote Address** | Responder's address. | none specified |
| **State** | State of the IKE security association:<br>• **Matured**—The IKE security association is established.<br>• **Not matured**—The IKE security association is in the process of negotiation. | none specified |

**Table 11: show ike security-associations Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Initiator cookie** | When the IKE negotiation is triggered, a random number is sent to the remote node. | All levels |
| **Responder cookie** | The remote node generates its own random number and sends it back to the initiator as a verification that the packets were received.<br><br>Of the numerous security services available, protection against denial of service (DoS) is one of the most difficult to address. A "cookie" or anticlogging token (ACT) is aimed at protecting the computing resources from attack without spending excessive CPU resources to determine the cookie's authenticity. An exchange prior to CPU-intensive public key operations can thwart some DoS attempts (such as simple flooding with invalid IP source addresses). | All levels |
| **Exchange type** | Specifies the number of messages in an IKE exchange, and the payload types that are contained in each message. Each exchange type provides a particular set of security services, such as anonymity of the participants, perfect forward secrecy of the keying material, and authentication of the participants. Junos OS supports two types of exchanges:<br><br>• **Main**—The exchange is done with six messages. **Main** encrypts the payload, protecting the identity of the neighbor.<br><br>• **Aggressive**—The exchange is done with three messages. **Aggressive** does not encrypt the payload, leaving the identity of the neighbor unprotected. | All Levels |
| **Authentication method** | Type of authentication determines which payloads are exchanged and when they are exchanged. The Junos OS supports only **pre-shared keys**. | **detail** |
| **Local** | Prefix and port number of the local end. | **detail** |
| **Remote** | Prefix and port number of the remote end. | **detail** |
| **Lifetime** | Number of seconds remaining until the IKE security association expires. | **detail** |

Table 11: show ike security-associations Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Algorithms** | Header for the IKE algorithms output.<br>• **Authentication**—Type of authentication algorithm used:**md5** or **sha1**.<br>• **Encryption**—Type of encryption algorithm used: **des-cbc**, **3des-cbc**, or **None**.<br>• **Pseudo random function**—Function that generates highly unpredictable random numbers:**hmac-md5** or**hmac-sha1**. | **detail** |
| **Traffic statistics** | Number of bytes and packets received and transmitted on the IKE security association.<br>• **Input bytes, Output bytes**—Number of bytes received and transmitted on the IKE security association.<br>• **Input packets, Output packets**—Number of packets received and transmitted on the IKE security association. | **detail** |
| **Flags** | Notification to the key management process of the status of the IKE negotiation:<br>• **caller notification sent**—Caller program notified about the completion of the IKE negotiation.<br>• **waiting for done**—Negotiation is done. The library is waiting for the remote end retransmission timers to expire.<br>• **waiting for remove**—Negotiation has failed. The library is waiting for the remote end retransmission timers to expire before removing this negotiation.<br>• **waiting for policy manager**—Negotiation is waiting for a response from the policy manager. | **detail** |
| **IPsec security associates** | Number of IPsec security associations created and deleted with this IKE security association. | **detail** |

Table 11: show ike security-associations Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| Phase 2 negotiations in progress | Number of phase 2 IKE negotiations in progress and status information:<br>• **Negotiation type**—Type of phase 2 negotiation. The Junos OS currently supports **quick mode**.<br>• **Message ID**—Unique identifier for a phase 2 negotiation.<br>• **Local identity**—Identity of the local phase 2 negotiation. The format is *id-type-name* (*proto-name*:*port-number*,[0..*id-data-len*] = *iddata-presentation*)<br>• **Remote identity**—Identity of the remote phase 2 negotiation. The format is *id-type-name* (*proto-name*:*port-number*,[0..*id-data-len*] = *iddata-presentation*)<br>• **Flags**—Notification to the key management process of the status of the IKE negotiation:<br>  • **caller notification sent**—Caller program notified about the completion of the IKE negotiation.<br>  • **waiting for done**—Negotiation is done. The library is waiting for the remote end retransmission timers to expire.<br>  • **waiting for remove**—Negotiation has failed. The library is waiting for the remote end retransmission timers to expire before removing this negotiation.<br>  • **waiting for policy manager**—Negotiation is waiting for a response from the policy manager. | detail |

## Sample Output

### show ike security-associations

```
user@host> show ike security-associations
Remote Address  State        Initiator cookie  Responder cookie  Exchange type
192.0.2.4         Matured       93870456fa000011  723a20713700003e  Main
```

### show ike security-associations detail

```
user@host> show ike security-associations detail
IKE peer 192.0.2.4
  Role: Initiator, State: Matured
  Initiator cookie: cf22bd81a7000001, Responder cookie: fe83795c2800002e
  Exchange type: Main, Authentication method: Pre-shared-keys
  Local: 192.0.2.5:500, Remote: 192.0.2.4:500
  Lifetime: Expires in 187 seconds
  Algorithms:
   Authentication        : md5
   Encryption            : 3des-cbc
   Pseudo random function: hmac-md5
  Traffic statistics:
   Input  bytes  :                1000
   Output bytes  :                1280
   Input  packets:                   5
   Output packets:                   9
  Flags: Caller notification sent
  IPsec security associations: 2 created, 0 deleted
  Phase 2 negotiations in progress: 1

Negotiation type: Quick mode, Role: Initiator, Message ID: 3582889153
    Local: 192.0.2.5:500, Remote: 192.0.2.4:500
    Local identity: ipv4_subnet(tcp:80,[0..7]=10.1.1.0/24)
    Remote identity: ipv4_subnet(tcp:100,[0..7]=10.1.2.0/24)
    Flags: Caller notification sent, Waiting for done
```

## Release Information

Command introduced before Junos OS Release 7.4.

#### RELATED DOCUMENTATION

clear ike security-associations | 278

# show interfaces (Encryption)

IN THIS SECTION



## Syntax

```
show interfaces es-fpc/pic/port:channel
<brief | detail | extensive | terse>
<descriptions>
<media>
<snmp-index snmp-index>
<statistics>
```

## Description

(M Series and T Series routers only) Display status information about the specified encryption interface.

## Options

| | |
|---|---|
| es-*fpc*/*pic*/*port*:*channel* | Display standard status information about the specified encryption interface. |

| | |
|---|---|
| **brief \| detail \| extensive \| terse** | (Optional) Display the specified level of output. |
| **descriptions** | (Optional) Display interface description strings. |
| **media** | (Optional) Display media-specific information about network interfaces. |
| **snmp-index** ***snmp-index*** | (Optional) Display information for the specified SNMP index of the interface. |
| **statistics** | (Optional) Display static interface statistics. |

## Required Privilege Level

view

## Output Fields

Table 12 on page 299 lists the output fields for the `show interfaces` (Encryption) command. Output fields are listed in the approximate order in which they appear.

**Table 12: Encryption show interfaces Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Physical Interface** | | |
| **Physical interface** | Name of the physical interface. | All levels |
| **Enabled** | State of the interface. Possible values are described in the “Enabled Field” section under *Common Output Fields Description*. | All levels |
| **Interface index** | Physical interface's index number, which reflects its initialization sequence. | **detail extensive** none |

**Table 12: Encryption show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **SNMP ifIndex** | SNMP index number for the physical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Type** | Encapsulation being used on the interface. | All levels |
| **Link-level type** | Encapsulation being used on the physical interface. | All levels |
| **MTU** | MTU size on the physical interface. | All levels |
| **Speed** | Speed at which the interface is running. | All levels |
| **Device flags** | Information about the physical device. Possible values are described in the “Link Flags” section under *Common Output Fields Description*. | All levels |
| **Interface flags** | Information about the interface. Possible values are described in the “Interface Flags” section under *Common Output Fields Description*. | All levels |
| **Input rate** | Input rate in bits per second (bps) and packets per second (pps). | None specified |
| **Output rate** | Output rate in bps and pps. | None specified |
| **Hold-times** | Current interface hold-time up and hold-time down, in milliseconds. | **detail extensive** |
| **Statistics last cleared** | Time when the statistics for the interface were last set to zero. | **detail extensive** |

**Table 12: Encryption show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Traffic statistics** | Number and rate of bytes and packets received and transmitted on the physical interface.<br>• **Input bytes, Output bytes**—Number of bytes received and transmitted on the interface.<br>• **Input packets, Output packets**—Number of packets received and transmitted on the interface.<br>• **Anti-replay failures**—Total number of antireplay failures seen on all tunnels configured on the ES PIC.<br>• **Authentication**—Total number of authentication failures seen on all tunnels configured on the ES PIC. | **detail extensive** |
| **Egress queues** | Total number of egress queues supported on the specified interface. | **detail extensive** |
| **Queue counters** | CoS queue number and its associated user-configured forwarding class name.<br>• **Queued packets**—Number of queued packets.<br>• **Transmitted packets**—Number of transmitted packets.<br>• **Dropped packets**—Number of packets dropped by the ASIC's RED mechanism. | **detail extensive** |
| **Logical Interface** | | |
| **Logical interface** | Name of the logical interface. | All levels |
| **Index** | Logical interface index number, which reflects its initialization sequence. | **detail extensive** none |
| **SNMP ifIndex** | Logical interface SNMP interface index number. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |

**Table 12: Encryption show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Flags** | Information about the logical interface. Possible values are described in the “Logical Interface Flags” section under *Common Output Fields Description*. | All levels |
| **IP-Header** | IP header of the logical interface. | All levels |
| **Encapsulation** | Encapsulation on the logical interface. | All levels |
| *protocol-family* | Protocol family configured on the logical interface. If the protocol is **inet**, the IP address of the interface is also displayed. | **brief** |
| **Input packets** | Number of packets received on the logical interface. | None specified |
| **Output packets** | Number of packets transmitted on the logical interface. | None specified |
| **Traffic statistics** | Total number of bytes and packets received and transmitted on the logical interface. These statistics are the sum of the local and transit statistics. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |
| **Local statistics** | Statistics for traffic received from and transmitted to the Routing Engine. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |
| **Transit statistics** | Statistics for traffic transiting the router. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |
| **Protocol** | Protocol family configured on the logical interface, such as **iso**, **inet6**, **mpls**. | **detail extensive** none |

**Table 12: Encryption show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **MTU** | MTU size on the logical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Route table** | Routing table in which the logical interface address is located. For example, **0** refers to the routing table inet.0. | **detail extensive** |
| **Flags** | Information about the protocol family flags. Possible values are described in the “Family Flags” section under *Common Output Fields Description*. | **detail extensive** none |
| **Addresses, Flags** | Information about the address flags. Possible values are described in the “Addresses Flags” section under *Common Output Fields Description*.Address | **detail extensive** none |
| **Destination** | IP address of the remote side of the connection. | **detail extensive** none |
| **Local** | IP address of the logical interface. | **detail extensive** none |
| **Broadcast** | Broadcast address of the logical interface. | **detail extensive** |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |

## Sample Output

### show interfaces (Encryption)

```
user@host> show interfaces es-0/3/0
Physical interface: es-0/3/0, Enabled, Physical link is Up
  Interface index: 138, SNMP ifIndex: 71
  Type: IPSEC, Link-level type: IPSEC-over-IP, MTU: 3900, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)

  Logical interface es-0/3/0.0 (Index 70) (SNMP ifIndex 45)
    Flags: Hardware-Down Point-To-Point SNMP-Traps
    IP-Header 10.0.10.2:10.0.10.1::df:64:00000000 Encapsulation: IPSEC
    Input packets : 0
    Output packets: 0
    Protocol inet, MTU: 3800
      Flags: None
      Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
        Destination: 10.10.0.2, Local: 10.10.0.1
```

### show interfaces brief (Encryption)

```
user@host> show interfaces es-0/3/0 brief
Physical interface: es-0/3/0, Enabled, Physical link is Up
  Type: IPSEC, Link-level type: IPSEC-over-IP, MTU: 3900, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps

  Logical interface es-0/3/0.0
    Flags: Hardware-Down Point-To-Point SNMP-Traps
    IP-Header 10.0.10.2:10.0.10.1::df:64:00000000 Encapsulation: IPSEC
    inet  10.10.0.1        --> 10.10.0.2s
```

### show interfaces detail (Encryption)

```
user@host> show interfaces es-0/3/0 detail
Physical interface: es-0/3/0, Enabled, Physical link is Up
  Interface index: 138, SNMP ifIndex: 71, Generation: 21
  Type: IPSEC, Link-level type: IPSEC-over-IP, MTU: 3900, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                    0                    0 bps
   Output bytes  :                    0                    0 bps
   Input  packets:                    0                    0 pps
   Output packets:                    0                    0 pps
   Anti-replay failures       : 0
   Authentication failures    : 0
  Egress queues: 4 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0 best-effort                    0                    0                    0
    1 expedited-fo                   0                    0                    0
    2 assured-forw                   0                    0                    0
    3 network-cont                   0                    0                    0

  Logical interface es-0/3/0.0 (Index 70) (SNMP ifIndex 45) (Generation 9)
    Flags: Hardware-Down Point-To-Point SNMP-Traps
    IP-Header 10.0.10.2:10.0.10.1::df:64:00000000 Encapsulation: IPSEC
    Traffic statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Transit statistics:
     Input  bytes  :                    0                    0 bps
     Output bytes  :                    0                    0 bps
     Input  packets:                    0                    0 pps
     Output packets:                    0                    0 pps
```

```
    Protocol inet, MTU: 3800, Generation: 22, Route table: 0
      Flags: None
      Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
        Destination: 10.10.0.2, Local: 10.10.0.1, Broadcast: Unspecified,
        Generation: 26
```

### show interfaces extensive (Encryption)

```
user@host> show interfaces es-0/3/0 extensive
Physical interface: es-0/3/0, Enabled, Physical link is Up
  Interface index: 138, SNMP ifIndex: 71, Generation: 21
  Type: IPSEC, Link-level type: IPSEC-over-IP, MTU: 3900, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                    0                    0 bps
   Output bytes  :                    0                    0 bps
   Input  packets:                    0                    0 pps
   Output packets:                    0                    0 pps
   Anti-replay failures       : 0
   Authentication failures    : 0
  Egress queues: 4 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0 best-effort                   0                    0                    0
    1 expedited-fo                  0                    0                    0
    2 assured-forw                  0                    0                    0
    3 network-cont                  0                    0                    0

  Logical interface es-0/3/0.0 (Index 70) (SNMP ifIndex 45) (Generation 9)
    Flags: Hardware-Down Point-To-Point SNMP-Traps
    IP-Header 10.0.10.2:10.0.10.1::df:64:00000000 Encapsulation: IPSEC
    Traffic statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
```

```
      Input  packets:                   0
      Output packets:                   0
     Transit statistics:
      Input  bytes  :                   0                    0 bps
      Output bytes  :                   0                    0 bps
      Input  packets:                   0                    0 pps
      Output packets:                   0                    0 pps
     Protocol inet, MTU: 3800, Generation: 22, Route table: 0
       Flags: None
       Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
         Destination: 10.10.0.2, Local: 10.10.0.1, Broadcast: Unspecified,
         Generation: 26
```

## Release Information

Command introduced before Junos OS Release 7.4.

# show interfaces (GRE)

IN THIS SECTION

## Syntax

```
show interfaces interface-type
<brief | detail | extensive | terse>
<descriptions>
<media>
<snmp-index snmp-index>
<statistics>
```

## Description

Display status information about the specified generic routing encapsulation (GRE) interface.

## Options

| | |
|---|---|
| *interface-type* | On M Series and T Series routers and EX Series switches, the interface type is **gr-***fpc/pic/port*. |
| **brief \| detail \| extensive \| terse** | (Optional) Display the specified output level of interface information. |
| **descriptions** | (Optional) Display interface description strings. |
| **media** | (Optional) Display media-specific information about network interfaces. |
| **snmp-index** *snmp-index* | (Optional) Display information for the specified SNMP index of the interface. |
| **statistics** | (Optional) Display static interface statistics. |

**NOTE**: You can configure generic routing encapsulation (GRE) interfaces (gre-x/y/z) only for GMPLS control channels. GRE interfaces are not supported or configurable for other applications. For more information about GMPLS, see the MPLS Applications User Guide.

## Required Privilege Level

view

## Output Fields

Table 13 on page 309 lists the output fields for the `show interfaces` (GRE) command. Output fields are listed in the approximate order in which they appear.

**Table 13: GRE show interfaces Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Physical Interface** | | |
| **Physical interface** | Name of the physical interface. | All levels |
| **Enabled** | State of the interface. Possible values are described in the “Enabled Field” section under *Common Output Fields Description*. | All levels |
| **Interface index** | Physical interface's index number, which reflects its initialization sequence. | **detail extensive** none |
| **SNMP ifIndex** | SNMP index number for the physical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Type** | Type of interface. | All levels |
| **Link-level type** | Encapsulation used on the physical interface. | All levels |
| **MTU** | MTU size on the physical interface. | All levels |

Table 13: GRE show interfaces Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Speed** | Speed at which the interface is running. | All levels |
| **Hold-times** | Current interface hold-time up and hold-time down, in milliseconds. | **detail extensive** |
| **Device Flags** | Information about the physical device. Possible values are described in the "Device Flags" section under *Common Output Fields Description*. | All levels |
| **Interface Flags** | Information about the interface. Possible values are described in the "Interface Flags" section under *Common Output Fields Description*. | All levels |
| **Input rate** | Input rate in bits per second (bps) and packets per second (pps). | None specified |
| **Output rate** | Output rate in bps and pps. | None specified |
| **Statistics last cleared** | Time when the statistics for the interface were last set to zero. | **detail extensive** |
| **Traffic statistics** | The number of and the rate at which input and output bytes and packets are received and transmitted on the physical interface.<br>• **Input bytes**—Number of bytes received on the interface.<br>• **Output bytes**—Number of bytes transmitted on the interface.<br>• **Input packets**—Number of packets received on the interface.<br>• **Output packets**—Number of packets transmitted on the interface. | **detail extensive** |
| **Logical Interface** | | |
| **Logical interface** | Name of the logical interface. | All levels |
| **Index** | Logical interface index number, which reflects its initialization sequence. | **detail extensive** none |

**Table 13: GRE show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **SNMP ifIndex** | Logical interface SNMP interface index number. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support. | **detail extensive** |
| **Flags** | Information about the logical interface. Possible values listed in the “Logical Interface Flags” section under *Common Output Fields Description*. describe general information about the logical interface.<br>GRE-specific information about the logical interface is indicated by the presence or absence of the following value in this field:<br>• **Reassemble-Pkts**—If the **Flags** field includes this string, the GRE tunnel is configured to reassemble tunnel packets that were fragmented after tunnel encapsulation. | All levels |
| **IP-Header** | IP header of the logical interface. If the `tunnel key` statement is configured, this information is included in the **IP Header** entry.<br>GRE-specific information about the logical interface is indicated by the presence or absence of the following value in this field:<br>• **df**—If the **IP-Header** field includes this string immediately following the 16 bits of identification information (that is, if **:df:** displays after the twelfth byte), the GRE tunnel is configured to allow fragmentation of GRE packets after encapsulation. | All levels |
| **Encapsulation** | Encapsulation on the logical interface. | All levels |
| `L2 Routing Instance` | Name of the Layer 2 routing instance associated with the GRE interface. | All levels |
| `L3 Routing Instance` | Name of the Layer 3 routing instance associated with the GRE interface. | All levels |

**Table 13: GRE show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| `Copy-tos-to-outer-ip-header` | Status of type of service (ToS) bits in the GRE packet header:<br>• `On`—ToS bits were copied from the payload packet header into the header of the IP packet sent through the GRE tunnel.<br>• `Off`—ToS bits were not copied from the payload packet header and are set to 0 in the GRE packet header.<br>**NOTE**: EX Series switches do not support copying ToS bits to the encapsulated packet, so the value of this field is always `Off` in switch output. | `detail extensive` |
| `Gre keepalives configured` | Indicates whether a GRE keepalive time and hold time are configured for the GRE tunnel.<br>**NOTE**: EX Series switches do not support configuration of GRE tunnel keepalive times and hold times, so the value of this field is always `Off` in switch output. | `detail extensive` |
| `Gre keepalives adjacency state` | Status of the other end of the GRE tunnel: `Up` or `Down`. If keepalive messages are not received by either end of the GRE tunnel within the hold-time period, the GRE keepalive adjacency state is down even when the GRE tunnel is up. | `detail extensive` |
| **Input packets** | Number of packets received on the logical interface. | None specified |
| **Output packets** | Number of packets transmitted on the logical interface. | None specified |
| **Traffic statistics** | Total number of bytes and packets received and transmitted on the logical interface. These statistics are the sum of the local and transit statistics. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize.<br>• **Input rate**—Rate of bits and packets received on the interface.<br>• **Output rate**—Rate of bits and packets transmitted on the interface. | **detail extensive** |

Table 13: GRE show interfaces Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Local statistics** | Statistics for traffic received from and transmitted to the Routing Engine. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |
| **Transit statistics** | Statistics for traffic transiting the router. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** none |
| **Protocol** | Protocol family configured on the logical interface, such as **iso**, **inet6**, or **mpls**. | **detail extensive none** |
| *protocol-family* | Protocol family configured on the logical interface. If the protocol is **inet**, the IP address of the interface is also displayed. | **brief** |
| **MTU** | MTU size on the logical interface. | **detail extensive none** |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Route table** | Routing table in which the logical interface address is located. For example, **0** refers to the routing table inet.0. | **detail extensive** |
| **Flags** | Information about the protocol family flags. Possible values are described in the “Family Flags” section under *Common Output Fields Description*. | **detail extensive** none |
| **Addresses, Flags** | Information about the address flags. Possible values are described in the “Addresses Flags” section under *Common Output Fields Description*. | **detail extensive** none |
| **Destination** | IP address of the remote side of the connection. | **detail extensive** none |

Table 13: GRE show interfaces Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Local** | IP address of the logical interface. | **detail extensive** none |
| **Broadcast** | Broadcast address of the logical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |

## Sample Output

### show interfaces (GRE)

```
user@host> show interfaces gr-1/2/0
Physical interface: gr-0/0/0, Enabled, Physical link is Up
  Interface index: 132, SNMP ifIndex: 26
  Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)

  Logical interface gr-0/0/0.0 (Index 68) (SNMP ifIndex 47)
    Flags: Point-To-Point SNMP-Traps 16384
    IP-Header 192.0.2.2:192.0.2.1:47:df:64:0000000000000000 Encapsulation: GRE-NULL
  Input packets : 0
  Output packets: 0
    Protocol inet, MTU: 1476
      Flags: None
      Addresses, Flags: Is-Primary
        Local: 198.51.100.1
```

### show interfaces brief (GRE)

```
user@host> show interfaces gr-1/2/0 brief
Physical interface: gr-1/2/0, Enabled, Physical link is Up
  Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps

  Logical interface gr-1/2/0.0
    Flags: Hardware-Down Point-To-Point SNMP-Traps 0x4000
    IP-Header 10.10.0.2:10.10.0.1:47:df:64:0000000000000000
    Encapsulation: GRE-NULL
    inet  10.100.0.1/30
    mpls
```

### show interfaces detail (GRE)

```
user@host> show interfaces gr-1/2/0 detail
Physical interface: gr-0/0/0, Enabled, Physical link is Up
  Interface index: 132, SNMP ifIndex: 26, Generation: 13
  Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                   0                    0 bps
   Output bytes  :                   0                    0 bps
   Input  packets:                   0                    0 pps
   Output packets:                   0                    0 pps

  Logical interface gr-0/0/0.0 (Index 68) (SNMP ifIndex 47) (Generation 8)
    Flags: Point-To-Point SNMP-Traps 16384
    IP-Header 192.0.2.2:192.0.2.1:47:df:64:0000000000000000 Encapsulation: GRE-NULL
    Traffic statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Local statistics:
```

```
      Input  bytes  :                  0
      Output bytes  :                  0
      Input  packets:                  0
      Output packets:                  0
     Transit statistics:
      Input  bytes  :                  0                    0 bps
      Output bytes  :                  0                    0 bps
      Input  packets:                  0                    0 pps
      Output packets:                  0                    0 pps
     Protocol inet, MTU: 1476, Generation: 12, Route table: 0
       Flags: None
       Addresses, Flags: Is-Primary
         Destination: Unspecified, Local: 198.51.100.1, Broadcast: Unspecified,
         Generation: 15
```

### show interfaces (Layer 2 Services Over GRE Interfaces)

```
user@host> show interfaces gr-2/2/10
show interfaces gr-2/2/10
Physical interface: gr-2/2/10, Enabled, Physical link is Up
  Interface index: 214, SNMP ifIndex: 690
  Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 1000mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)

  Logical interface gr-2/2/10.0 (Index 342) (SNMP ifIndex 10834)
    Flags: Up Point-To-Point SNMP-Traps 0x4000 IP-Header
203.0.113.1:203.0.113.254:47:df:64:0000000000000000 Encapsulation: GRE-NULL
    L2 Routing Instance: vs1, L3 Routing Instance: default
    Copy-tos-to-outer-ip-header: Off
    Gre keepalives configured: Off, Gre keepalives adjacency state: down
    Input packets : 2
    Output packets: 0
    Protocol bridge, MTU: 1476
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 6/8, Local: 6.0.0.1, Broadcast: 6.255.255.255
```

### show interfaces extensive (Layer 2 Services Over GRE Interfaces)

```
user@host> show interfaces gr-2/2/10.0 extensive

     Flags: SNMP-Traps Encapsulation: ENET2
     L2 Routing Instance: vs1, L3 Routing Instance: default
     Traffic statistics:
      Input  bytes  :           58851250
      Output bytes  :                  0
      Input  packets:            1279375
      Output packets:                  0
     Local statistics:
      Input  bytes  :                  0
      Output bytes  :                  0
      Input  packets:                  0
      Output packets:                  0
     Transit statistics:
      Input  bytes  :           58851250                75136 bps
      Output bytes  :                  0                    0 bps
      Input  packets:            1279375                  204 pps
      Output packets:                  0                    0 pps
     Protocol bridge, MTU: 1476, Generation: 175, Route table: 7
       Flags: Access-Mode
```

### show interfaces detail (GRE) on an EX4200 Virtual Chassis Member Switch

```
user@host> show interfaces gr-2/0/15 detail
Physical interface: gr-2/0/15, Enabled, Physical link is Up
  Interface index: 195, SNMP ifIndex: 846, Generation: 198
  Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 1000mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Current address: 00:00:5e:00:53:d2, Hardware address: 00:00:5e:00:53:d2
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Statistics last cleared: 2011-09-14 17:43:15 UTC (00:00:18 ago)
  Traffic statistics:
   Input  bytes  :             5600636                    0 bps
   Output bytes  :             5600636                    0 bps
   Input  packets:               20007                    0 pps
   Output packets:               20007                    0 pps
```

```
 IPv6 transit statistics:
  Input  bytes  :                   0
  Output bytes  :                   0
  Input  packets:                   0
  Output packets:                   0

Logical interface gr-2/0/15.0 (Index 75) (SNMP ifIndex 847) (HW Token 4093)
(Generation 140)
  Flags: Point-To-Point SNMP-Traps 0x0
  IP-Header 192.168.30.2:192.168.20.3:47:df:64:0000000000000000
  Encapsulation: GRE-NULL
  Copy-tos-to-outer-ip-header: Off
  Gre keepalives configured: Off, Gre keepalives adjacency state: down
  Traffic statistics:
   Input  bytes  :             5600886
   Output bytes  :             2881784
   Input  packets:               20010
   Output packets:               10018
  Local statistics:
   Input  bytes  :                 398
   Output bytes  :                 264
   Input  packets:                   5
   Output packets:                   3
  Transit statistics:
   Input  bytes  :             5600488                    0 bps
   Output bytes  :             2881520                    0 bps
   Input  packets:               20005                    0 pps
   Output packets:               10015                    0 pps
  Protocol inet, Generation: 159, Route table: 0
    Flags: None
    Addresses, Flags: Is-Preferred Is-Primary
      Destination: 10.10.10/8, Local: 10.10.10.10, Broadcast: 10.10.10.255,
      Generation: 144

Logical interface gr-2/0/15.1 (Index 80) (SNMP ifIndex 848) (HW Token 4088)
(Generation 150)
  Flags: Point-To-Point SNMP-Traps 0x0
  IP-Header 192.168.40.2:192.168.30.1:47:df:64:0000000000000000
  Encapsulation: GRE-NULL
  Copy-tos-to-outer-ip-header: Off
  Gre keepalives configured: Off, Gre keepalives adjacency state: down
  Traffic statistics:
   Input  bytes  :                 260
```

```
      Output bytes  :              2880148
      Input  packets:                    4
      Output packets:                10002
     Local statistics:
      Input  bytes  :                  112
      Output bytes  :                    0
      Input  packets:                    2
      Output packets:                    0
     Transit statistics:
      Input  bytes  :                  148                    0 bps
      Output bytes  :              2880148                    0 bps
      Input  packets:                    2                    0 pps
      Output packets:                10002                    0 pps
     Protocol inet, Generation: 171, Route table: 0
       Flags: None
       Addresses, Flags: Is-Preferred Is-Primary
         Destination: 10.10.10/8, Local: 10.10.10.10, Broadcast: 10.10.10.255,
         Generation: 160
```

### show interfaces extensive (GRE)

The output for the `show interfaces extensive` command is identical to that for the `show interfaces detail` command. For sample output, see "show interfaces detail (GRE)" on page 315 and "show interfaces detail (GRE) on an EX4200 Virtual Chassis Member Switch" on page 317.

### show interfaces gr-2/0/10 for GRE IPv6 tunnel

```
user@host> show interfaces gr-2/0/10
show interfaces gr-2/0/10
Physical interface: gr-2/0/10, Enabled, Physical link is Up
  Interface index: 140, SNMP ifIndex: 559
  Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 1000mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Input rate     : 4952 bps (3 pps)
  Output rate    : 200 bps (0 pps)

  Logical interface gr-2/0/10.0 (Index 355) (SNMP ifIndex 857)
    Flags: Up Point-To-Point SNMP-Traps 0x4000 IP-Header
1000::11:0:11:1-1000::11:2:13:2-47-64-0-0-0000000000000000 Encapsulation: GRE-NULL
    Copy-tos-to-outer-ip-header: Off, Copy-tos-to-outer-ip-header-transit: Off
```

```
Gre keepalives configured: Off, Gre keepalives adjacency state: down
Input packets : 60
Output packets: 83
Protocol inet, MTU: 9082
Max nh cache: 0, New hold nh limit: 0, Curr nh cnt: 0, Curr new hold cnt: 0, NH drop cnt: 0
  Flags: Sendbcast-pkt-to-re
  Addresses, Flags: Is-Preferred Is-Primary
    Destination: 14.0.13/24, Local: 14.0.13.1, Broadcast: 14.0.13.255
Protocol iso, MTU: 9082
Protocol inet6, MTU: 9082
Max nh cache: 0, New hold nh limit: 0, Curr nh cnt: 0, Curr new hold cnt: 0, NH drop cnt: 0
  Addresses, Flags: Is-Preferred Is-Primary
    Destination: 1400::14:0:13:0/120, Local: 1400::14:0:13:1
  Addresses, Flags: Is-Preferred
    Destination: fe80::/64, Local: fe80::2a0:a520:2875:4992
Protocol mpls, MTU: 9070, Maximum labels: 3
  Flags: Is-Primary
```

## Release Information

Command introduced before Junos OS Release 7.4.

Command introduced before Junos OS Release 17.3R1.

# show interfaces (IP-over-IP)

IN THIS SECTION

## Syntax

```
show interfaces interface-type
<brief | detail | extensive | terse>
<descriptions>
<media>
<snmp-index snmp-index>
<statistics>
```

## Description

Display status information about the specified IP-over-IP interface.

## Options

| | |
|---|---|
| *interface-type* | On M Series and T Series routers, the interface type is **ip-***fpc*/*pic*/*port*. |
| **brief \| detail \| extensive \| terse** | (Optional) Display the specified level of output. |
| **descriptions** | (Optional) Display interface description strings. |
| **media** | (Optional) Display media-specific information about network interfaces. |
| **snmp-index** *snmp-index* | (Optional) Display information for the specified SNMP index of the interface. |
| **statistics** | (Optional) Display static interface statistics. |

## Required Privilege Level

view

## Output Fields

Table 14 on page 322 lists the output fields for the `show interfaces` (IP-over-IP) command. Output fields are listed in the approximate order in which they appear.

**Table 14: IP-over-IP show interfaces Output Fields**

| Field | Field Description | Level of Output |
|---|---|---|
| **Physical Interface** | | |
| **Physical interface** | Name of the physical interface. | All levels |
| **Enabled** | State of the interface. Possible values are described in the “Enabled Field” section under *Common Output Fields Description*. | All levels |
| **Interface index** | Physical interface's index number, which reflects its initialization sequence. | **detail extensive** none |
| **SNMP ifIndex** | SNMP index number for the physical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Type** | Type of interface. | All levels |
| **Link-level type** | Encapsulation used on the physical interface. | All levels |
| **MTU** | MTU size on the physical interface. | All levels |

Table 14: IP-over-IP show interfaces Output Fields *(Continued)*

| Field | Field Description | Level of Output |
|---|---|---|
| **Speed** | Speed at which the interface is running. | All levels |
| **Hold-times** | Current interface hold-time up and hold-time down, in milliseconds. | **detail extensive** |
| **Device flags** | Information about the physical device. Possible values are described in the “Device Flags” section under *Common Output Fields Description*. | All levels |
| **Interface flags** | Information about the interface. Possible values are described in the “Interface Flags” section under *Common Output Fields Description*. | All levels |
| **Input rate** | Input rate in bits per second (bps) and packets per second (pps). | None specified |
| **Output rate** | Output rate in bps and pps. | None specified |
| **Statistics last cleared** | Time when the statistics for the interface were last set to zero. | **detail extensive** |
| **Traffic statistics** | Number and rate of bytes and packets received and transmitted on the physical interface.<br>• **Input bytes**—Number of bytes received on the interface.<br>• **Output bytes**—Number of bytes transmitted on the interface.<br>• **Input packets**—Number of packets received on the interface.<br>• **Output packets**—Number of packets transmitted on the interface. | **detail extensive** |
| **Logical Interface** | | |
| **Logical interface** | Name of the logical interface. | All levels |
| **Index** | Logical interface index number, which reflects its initialization sequence. | **detail extensive** none |

**Table 14: IP-over-IP show interfaces Output Fields *(Continued)***

| Field | Field Description | Level of Output |
|---|---|---|
| **SNMP ifIndex** | Logical interface SNMP interface index number. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support. | **detail extensive** |
| **Flags** | Information about the logical interface. Possible values are described in the "Logical Interface Flags" section under *Common Output Fields Description*. | All levels |
| **IP Header** | IP header of the logical interface. | All levels |
| **Encapsulation** | Encapsulation on the logical interface. | All levels |
| **Input packets** | Number of packets received on the logical interface. | None specified |
| **Output packets** | Number of packets transmitted on the logical interface. | None specified |
| **Traffic statistics** | Total number of bytes and packets received and transmitted on the logical interface. These statistics are the sum of the local and transit statistics. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize.<br>• **Input rate**—Rate of bits and packets received on the interface.<br>• **Output rate**—Rate of bits and packets transmitted on the interface. | **detail extensive** |
| **Local statistics** | Statistics for traffic received from and transmitted to the Routing Engine. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |

**Table 14: IP-over-IP show interfaces Output Fields *(Continued)***

| Field | Field Description | Level of Output |
|---|---|---|
| **Transit statistics** | Statistics for traffic transiting the router. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |
| **Protocol** | Protocol family configured on the logical interface, such as **iso**, **inet6**, or **mpls**. | **detail extensive none** |
| ***protocol-family*** | Protocol family configured on the logical interface. If the protocol is **inet**, the IP address of the interface is also displayed. | **brief** |
| **MTU** | MTU size on the logical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Route table** | Routing table in which the logical interface address is located. For example, **0** refers to the routing table inet.0. | **detail extensive** |
| **Flags** | Information about the protocol family flags. Possible values are described in the “Family Flags” section under *Common Output Fields Description*. | **detail extensive** none |

## Sample Output

### show interfaces (IP-over-IP)

```
user@host> show interfaces ip-0/0/0
Physical interface: ip-0/0/0, Enabled, Physical link is Up
  Interface index: 133, SNMP ifIndex: 27
  Type: IPIP, Link-level type: IP-over-IP, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
```

```
  Interface flags: SNMP-Traps
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)

  Logical interface ip-0/0/0.0 (Index 69) (SNMP ifIndex 49)
    Flags: Point-To-Point SNMP-Traps 16384
    IP-Header 192.0.2.1:192.0.2.2:4:df:64:00000000 Encapsulation: IPv4-NULL
  Input packets : 0
  Output packets: 0
    Protocol inet, MTU: 1480
      Flags: None
```

### show interfaces brief (IP-over-IP)

```
user@host> show interfaces ip-0/0/0 brief
Physical interface: ip-0/0/0, Enabled, Physical link is Up
  Type: IPIP, Link-level type: IP-over-IP, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: SNMP-Traps

  Logical interface ip-0/0/0.0
    Flags: Point-To-Point SNMP-Traps 16384
    IP-Header 192.0.2.1:192.0.2.2:4:df:64:00000000 Encapsulation: IPv4-NULL
    inet
```

### show interfaces detail (IP-over-iP)

```
user@host> show interfaces ip-0/0/0 detail
Physical interface: ip-0/0/0, Enabled, Physical link is Up
  Interface index: 133, SNMP ifIndex: 27, Generation: 14
  Type: IPIP, Link-level type: IP-over-IP, MTU: Unlimited, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                   0                    0 bps
   Output bytes  :                   0                    0 bps
   Input  packets:                   0                    0 pps
   Output packets:                   0                    0 pps
```

```
Logical interface ip-0/0/0.0 (Index 69) (SNMP ifIndex 49) (Generation 9)
  Flags: Point-To-Point SNMP-Traps 16384
  IP-Header 192.0.2.1:192.0.2.2:4:df:64:00000000 Encapsulation: IPv4-NULL
  Traffic statistics:
   Input  bytes  :                    0
   Output bytes  :                    0
   Input  packets:                    0
   Output packets:                    0
  Local statistics:
   Input  bytes  :                    0
   Output bytes  :                    0
   Input  packets:                    0
   Output packets:                    0
  Transit statistics:
   Input  bytes  :                    0                   0 bps
   Output bytes  :                    0                   0 bps
   Input  packets:                    0                   0 pps
   Output packets:                    0                   0 pps
  Protocol inet, MTU: 1480, Generation: 13, Route table: 0
    Flags: None
```

### show interfaces extensive (IP-over-IP)

The output for the show interfaces extensive command is identical to that for the show interfaces detail command. For sample output, see "show interfaces detail (IP-over-iP)" on page 326.

## Release Information

Command introduced before Junos OS Release 7.4.

# show interfaces (Logical Tunnel)

IN THIS SECTION



## Syntax

```
show interfaces interface-type
<brief | detail | extensive | terse>
<descriptions>
<media>
<snmp-index snmp-index>
<statistics>
```

## Description

Display status information about the specified logical tunnel interface.

## Options

*interface-type* On M Series and T Series routers, the interface type is **lt-***fpc*/*pic*/*port*.

| | |
|---|---|
| **brief \| detail \| extensive \| terse** | (Optional) Display the specified level of output. |
| **descriptions** | (Optional) Display interface description strings. |
| **media** | (Optional) Display media-specific information about network interfaces. |
| **snmp-index** *snmp-index* | (Optional) Display information for the specified SNMP index of the interface. |
| **statistics** | (Optional) Display static interface statistics. |

## Required Privilege Level

view

## Output Fields

Table 15 on page 329 lists the output fields for the `show interfaces` (logical tunnel) command. Output fields are listed in the approximate order in which they appear.

**Table 15: Logical Tunnel show interfaces Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Physical Interface** | | |
| **Physical interface** | Name of the physical interface. | All levels |
| **Enabled** | State of the interface. Possible values are described in the “Enabled Field” section under *Common Output Fields Description*. | All levels |
| **Interface index** | Physical interface index number, which reflects its initialization sequence. | **detail extensive** none |

**Table 15: Logical Tunnel show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **SNMP ifIndex** | SNMP index number for the physical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Type** | Type of interface. **Software-Pseudo** indicates a standard software interface with no associated hardware device. | All levels |
| **Link-level type** | Encapsulation used on the physical interface. | All levels |
| **MTU** | MTU size on the physical interface. | All levels |
| **Clocking** | Reference clock source: **Internal or External** when configured. Otherwise, **Unspecified.** | All levels |
| **Speed** | Speed at which the interface is running. | All levels |
| **Device flags** | Information about the physical device. Possible values are described in the "Device Flags" section under *Common Output Fields Description*. | All levels |
| **Interface flags** | Information about the interface. Possible values are described in the "Interface Flags" section under *Common Output Fields Description*. | All levels |
| **Link type** | Type of link. | All levels |
| **Link flags** | Information about the link. Possible values are described in the "Link Flags" section under *Common Output Fields Description*. | All levels |
| **Physical info** | Information about the physical interface. | All levels |
| **Hold-times** | Current interface hold-time up and hold-time down, in milliseconds. | **detail extensive** |

**Table 15: Logical Tunnel show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Current address** | Configured MAC address. | **detail extensive** none |
| **Hardware address** | Hardware MAC address. | **detail extensive** none |
| **Alternate link address** | Backup link address. | **detail extensive** none |
| **Last flapped** | Date, time, and how long ago the interface went from down to up. The format is **Last flapped:** ***year-month-day hour:minute:second timezone*** **(*hour:minute:second* ago)**. For example, **Last flapped: 2002-04-26 10:52:40 PDT (04:33:20 ago)**. | **detail extensive** none |
| **Statistics last cleared** | Time when the statistics for the interface were last set to zero. | **detail extensive** |
| **Traffic statistics** | Number and rate of bytes and packets received and transmitted on the physical interface.<br>• **Input bytes, Output bytes**—Number of bytes received and transmitted on the interface.<br>• **Input packets, Output packets**—Number of packets received and transmitted on the interface. | **detail extensive** |

**Table 15: Logical Tunnel show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Input errors** | Input errors on the interface. The following paragraphs explain the counters whose meaning might not be obvious:<br>• **Errors**—Sum of the incoming frame terminates and FCS errors.<br>• **Drops**—Number of packets dropped by the input queue of the I/O Manager ASIC. If the interface is saturated, this number increments once for every packet that is dropped by the ASIC's RED mechanism.<br>• **Framing errors**—Number of packets received with an invalid frame checksum (FCS).<br>• **Runts**—Number of frames received that are smaller than the runt threshold.<br>• **Giants**—Number of frames received that are larger than the giant threshold.<br>• **Policed discards**—Number of frames that the incoming packet match code discarded because they were not recognized or not of interest. Usually, this field reports protocols that the Junos OS does not handle.<br>• **Resource errors**—Sum of transmit drops. | **extensive** |

**Table 15: Logical Tunnel show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Output errors** | Output errors on the interface. The following paragraphs explain the counters whose meaning might not be obvious:<br>• **Carrier transitions**—Number of times the interface has gone from **down** to **up**. This number does not normally increment quickly, increasing only when the cable is unplugged, the far-end system is powered down and then up, or another problem occurs. If the number of carrier transitions increments quickly (perhaps once every 10 seconds), the cable, the far-end system, or the PIC is malfunctioning.<br>• **Errors**—Sum of the outgoing frame terminates and FCS errors.<br>• **Drops**—Number of packets dropped by the output queue of the I/O Manager ASIC. If the interface is saturated, this number increments once for every packet that is dropped by the ASIC's RED mechanism.<br>• **MTU errors**—Number of packets larger than the MTU threshold.<br>• **Resource errors**—Sum of transmit drops. | **extensive** |
| **Logical Interface** | | |
| **Logical interface** | Name of the logical interface. | All levels |
| **Index** | Logical interface index number, which reflects its initialization sequence. | **detail extensive** none |
| **SNMP ifIndex** | SNMP interface index number. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Flags** | Information about the logical interface. Possible values are described in the "Logical Interface Flags" section under *Common Output Fields Description*. | All levels |

**Table 15: Logical Tunnel show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Encapsulation** | Encapsulation on the logical interface. | All levels |
| **Traffic statistics** | Total number of bytes and packets received and transmitted on the logical interface. These statistics are the sum of the local and transit statistics. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize.<br>• **Input bytes**—Rate of bytes received on the interface.<br>• **Output bytes**—Rate of bytes transmitted on the interface.<br>• **Input packets**—Rate of packets received on the interface.<br>• **Output packets**—Rate of packets transmitted on the interface. | **detail extensive** |
| **Local statistics** | Statistics for traffic received from and transmitted to the Routing Engine. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |
| **Transit statistics** | Statistics for traffic transiting the router. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |
| **Protocol** | Protocol family configured on the logical interface, such as **iso**, **inet6**, **mpls**. | **detail extensive** none |
| **MTU** | MTU size on the logical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Route table** | Route table in which this address exists. For example, **Route table:0** refers to inet.0. | **detail extensive** |

**Table 15: Logical Tunnel show interfaces Output Fields** ***(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Flags** | Information about the protocol family flags. Possible values are described in the “Family Flags” section under *Common Output Fields Description*. | **detail extensive** none |
| **Addresses, Flags** | Information about the address flags. Possible values are described in the “Addresses Flags” section under *Common Output Fields Description*. | **detail extensive** none |
| **Destination** | IP address of the remote side of the connection. | **detail extensive** none |
| **Local** | IP address of the logical interface. | **detail extensive** none |
| **Broadcast** | Broadcast address of the logical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |

## Sample Output

### show interfaces extensive (Logical Tunnel)

```
user@host> show interfaces lt-1/0/0 extensive
Physical interface: lt-1/0/0, Enabled, Physical link is Up
  Interface index: 143, SNMP ifIndex: 70, Generation: 26
  Type: Logical-tunnel, Link-level type: Logical-tunnel, MTU: 0,
  Clocking: Unspecified, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Link type      : Unspecified
  Link flags     : None
  Physical info  : 13
```

```
Hold-times     : Up 0 ms, Down 0 ms
Current address: 00:00:5e:00:53:7e, Hardware address: Unspecified
Alternate link address: Unspecified
Last flapped   : 2004-03-03 15:53:52 PST (22:08:46 ago)
Statistics last cleared: Never
Traffic statistics:
 Input  bytes  :                    0                    0 bps
 Output bytes  :                    0                    0 bps
 Input  packets:                    0                    0 pps
 Output packets:                    0                    0 pps
Input errors:
  Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0,
  Policed discards: 0
Output errors:
  Carrier transitions: 1, Errors: 0, Drops: 0, MTU errors: 0

Logical interface lt-1/0/0.0 (Index 66) (SNMP ifIndex 467) (Generation 3024)
  Flags: Point-To-Point SNMP-Traps 16384 DLCI 100 Encapsulation: FR-NLPID
  Traffic statistics:
   Input  bytes  :                    0
   Output bytes  :                    0
   Input  packets:                    0
   Output packets:                    0
  Local statistics:
   Input  bytes  :                    0
   Output bytes  :                    0
   Input  packets:                    0
   Output packets:                    0
  Transit statistics:
   Input  bytes  :                    0                    0 bps
   Output bytes  :                    0                    0 bps
   Input  packets:                    0                    0 pps
   Output packets:                    0                    0 pps
  Protocol inet, MTU: 4470, Generation: 7034, Route table: 0
    Flags: None
    Addresses, Flags: Is-Preferred Is-Primary
      Destination: 10.1.1/24, Local: 10.1.1.1, Broadcast: Unspecified,
      Generation: 2054
```

## Release Information

Command introduced before Junos OS Release 7.4.

# show interfaces (Multicast Tunnel)

IN THIS SECTION



## Syntax

```
show interfaces interface-type
<brief | detail | extensive | terse>
<descriptions>
<media>
<snmp-index snmp-index>
<statistics>
```

## Description

Display status information about the specified multicast tunnel interface and its logical encapsulation and de-encapsulation interfaces.

## Options

| | |
|---|---|
| *interface-type* | On M Series and T Series routers, the interface type is **mt-***fpc*/*pic*/*port*. |
| **brief \| detail \| extensive \| terse** | (Optional) Display the specified level of output. |
| **descriptions** | (Optional) Display interface description strings. |
| **media** | (Optional) Display media-specific information about network interfaces. |
| **snmp-index** *snmp-index* | (Optional) Display information for the specified SNMP index of the interface. |
| **statistics** | (Optional) Display static interface statistics. |

## Additional Information

The multicast tunnel interface has two logical interfaces: encapsulation and de-encapsulation. These interfaces are automatically created by the Junos OS for every multicast-enabled VPN routing and forwarding (VRF) instance. The encapsulation interface carries multicast traffic traveling from the edge interface to the core interface. The de-encapsulation interface carries traffic coming from the core interface to the edge interface.

## Required Privilege Level

view

## Output Fields

Table 16 on page 339 lists the output fields for the `show interfaces` (Multicast Tunnel) command. Output fields are listed in the approximate order in which they appear.

**Table 16: Multicast Tunnel show interfaces Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Physical Interface** | | |
| **Physical interface** | Name of the physical interface. | All levels |
| **Enabled** | State of the interface. Possible values are described in the “Enabled Field” section under Common Output Fields Description. | All levels |
| **Interface index** | Physical interface's index number, which reflects its initialization sequence. | **detail extensive** none |
| **SNMP ifIndex** | SNMP index number for the physical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Type** | Type of interface. | All levels |
| **Link-level type** | Encapsulation used on the physical interface. | All levels |
| **MTU** | MTU size on the physical interface. | All levels |
| **Speed** | Speed at which the interface is running. | All levels |
| **Hold-times** | Current interface hold-time up and hold-time down, in milliseconds. | **detail extensive** |
| **Device flags** | Information about the physical device. Possible values are described in the “Device Flags” section under Common Output Fields Description. | All levels |

Table 16: Multicast Tunnel show interfaces Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Interface flags** | Information about the interface. Possible values are described in the “Interface Flags” section under Common Output Fields Description. | All levels |
| **Input Rate** | Input rate in bits per second (bps) and packets per second (pps). | None specified |
| **Output Rate** | Output rate in bps and pps. | None specified |
| **Statistics last cleared** | Time when the statistics for the interface were last set to zero. | **detail extensive** |
| **Traffic statistics** | Number and rate of bytes and packets received and transmitted on the physical interface.<br>• **Input bytes**—Number of bytes received on the interface.<br>• **Output bytes**—Number of bytes transmitted on the interface.<br>• **Input packets**—Number of packets received on the interface.<br>• **Output packets**—Number of packets transmitted on the interface. | All levels |

## Sample Output

### show interfaces (Multicast Tunnel)

```
user@host> show interfaces mt-1/2/0
Physical interface: mt-1/2/0, Enabled, Physical link is Up
  Interface index: 145, SNMP ifIndex: 41
  Type: Multicast-GRE, Link-level type: GRE, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
```

### show interfaces brief (Multicast Tunnel)

```
user@host> show interfaces mt-1/2/0 brief
Physical interface: mt-1/2/0, Enabled, Physical link is Up
  Type: Multicast-GRE, Link-level type: GRE, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: SNMP-Traps
```

### show interfaces detail (Multicast Tunnel)

```
user@host> show interfaces mt-1/2/0 detail
Physical interface: mt-1/2/0, Enabled, Physical link is Up
  Interface index: 145, SNMP ifIndex: 41, Generation: 28
  Type: Multicast-GRE, Link-level type: GRE, MTU: Unlimited, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Statistics last cleared: Never
   Traffic statistics:
    Input  bytes  :           170664562              560000 bps
    Output bytes  :           112345376              368176 bps
    Input  packets:             2439107                1000 pps
    Output packets:             2439120                1000 pps
```

### show interfaces extensive (Multicast Tunnel)

```
user@host> show interfaces mt-1/2/0 extensive
Physical interface: mt-1/2/0, Enabled, Physical link is Up
   Interface index: 141, SNMP ifIndex: 529, Generation: 144
   Type: Multicast-GRE, Link-level type: GRE, MTU: Unlimited, Speed: 800mbps
   Hold-times     : Up 0 ms, Down 0 ms
   Device flags   : Present Running
   Interface flags: SNMP-Traps
   Statistics last cleared: Never
   Traffic statistics:
    Input  bytes  :           170664562              560000 bps
    Output bytes  :           112345376              368176 bps
    Input  packets:             2439107                1000 pps
    Output packets:             2439120                1000 pps
```

```
      IPv6 transit statistics:
       Input  bytes  :                   0
       Output bytes  :                   0
       Input  packets:                   0
       Output packets:                   0

    Logical interface mt-1/2/0.32768 (Index 83) (SNMP ifIndex 556) (Generation 148)
      Flags: Point-To-Point SNMP-Traps 0x4000 IP-Header
192.0.2.1:10.0.0.6:47:df:64:0000000800000000 Encapsulation: GRE-NULL
      Traffic statistics:
       Input  bytes  :           170418430
       Output bytes  :           112070294
       Input  packets:             2434549
       Output packets:             2435593
       IPv6 transit statistics:
        Input  bytes  :                  0
        Output bytes  :                  0
        Input  packets:                  0
        Output packets:                  0
      Local statistics:
       Input  bytes  :                   0
       Output bytes  :               80442
       Input  packets:                   0
       Output packets:                1031
      Transit statistics:
       Input  bytes  :           170418430              560000 bps
       Output bytes  :           111989852              368176 bps
       Input  packets:             2434549                1000 pps
       Output packets:             2434562                1000 pps
       IPv6 transit statistics:
        Input  bytes  :                  0
        Output bytes  :                  0
        Input  packets:                  0
        Output packets:                  0
      Protocol inet, MTU: 1572, Generation: 182, Route table: 4
        Flags: None
      Protocol inet6, MTU: 1572, Generation: 183, Route table: 4
        Flags: None

    Logical interface mt-1/2/0.1081344 (Index 84) (SNMP ifIndex 560) (Generation 149)
      Flags: Point-To-Point SNMP-Traps 0x6000 Encapsulation: GRE-NULL
      Traffic statistics:
       Input  bytes  :              246132
```

```
        Output bytes  :              355524
        Input  packets:                4558
        Output packets:                4558
        IPv6 transit statistics:
         Input  bytes  :                  0
         Output bytes  :                  0
         Input  packets:                  0
         Output packets:                  0
       Local statistics:
        Input  bytes  :              246132
        Output bytes  :                   0
        Input  packets:                4558
        Output packets:                   0
       Transit statistics:
        Input  bytes  :                   0                    0 bps
        Output bytes  :              355524                    0 bps
        Input  packets:                   0                    0 pps
        Output packets:                4558                    0 pps
        IPv6 transit statistics:
         Input  bytes  :                  0
         Output bytes  :                  0
         Input  packets:                  0
         Output packets:                  0
       Protocol inet, MTU: Unlimited, Generation: 184, Route table: 4
         Flags: None
       Protocol inet6, MTU: Unlimited, Generation: 185, Route table: 4
         Flags: None
```

### show interfaces (Multicast Tunnel Encapsulation)

```
user@host> show interfaces mt-3/1/0.32768
Logical interface mt-3/1/0.32768 (Index 67) (SNMP ifIndex 0)
    Flags: Point-To-Point SNMP-Traps 0x4000
    IP-Header 198.51.100.1:10.255.70.15:47:df:64:0000000800000000
    Encapsulation: GRE-NULL
  Input packets : 0
  Output packets: 2
    Protocol inet, MTU: Unlimited
      Flags: None
```

### show interfaces (Multicast Tunnel De-Encapsulation)

```
user@host> show interfaces mt-3/1/0.49152
Logical interface mt-3/1/0.49152 (Index 74) (SNMP ifIndex 0)
    Flags: Point-To-Point SNMP-Traps 0x6000 Encapsulation: GRE-NULL
  Input packets : 0
  Output packets: 2
    Protocol inet, MTU: Unlimited
      Flags: None
```

## Release Information

Command introduced before Junos OS Release 7.4.

# show interfaces (PIM)

IN THIS SECTION

## Syntax

```
show interfaces interface-type
<brief | detail | extensive | terse>
<descriptions>
<media>
<snmp-index snmp-index>
<statistics>
```

## Description

Display status information about the specified Protocol Independent Multicast (PIM) de-encapsulation or PIM encapsulation interface, respectively.

## Options

| | |
|---|---|
| *interface-type* | On M Series and T Series routers, the PIM de-encapsulation interface type is **pd-***fpc*/*pic*/*port* and the PIM encapsulation interface type is **pe-***fpc*/*pic*/*port*. |
| **brief \| detail \| extensive \| terse** | (Optional) Display the specified level of output. |
| **descriptions** | (Optional) Display interface description strings. |
| **media** | (Optional) Display media-specific information about network interfaces. |
| **snmp-index** *snmp-index* | (Optional) Display information for the specified SNMP index of the interface. |
| **statistics** | (Optional) Display static interface statistics. |

## Required Privilege Level

view

## Output Fields

Table 17 on page 346 lists the output fields for the `show interfaces` (PIM de-encapsulation or encapsulation) command. Output fields are listed in the approximate order in which they appear.

**Table 17: PIM show interfaces Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Physical Interface** | | |
| **Physical interface** | Name of the physical interface. | All levels |
| **Enabled** | State of the interface. Possible values are described in the “Enabled Field” section under *Common Output Fields Description*. | All levels |
| **Interface index** | Physical interface's index number, which reflects its initialization sequence. | **detail extensive** none |
| **SNMP ifIndex** | SNMP index number for the physical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Type** | Type of interface. | All levels |
| **Link-level type** | Encapsulation used on the physical interface. | All levels |
| **MTU** | MTU size on the physical interface. | All levels |
| **Speed** | Speed at which the interface is running. | All levels |
| **Hold-times** | Current interface hold-time up and hold-time down, in milliseconds. | **detail extensive** |
| **Device flags** | Information about the physical device. Possible values are described in the “Device Flags” section under *Common Output Fields Description*. | All levels |

Table 17: PIM show interfaces Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Interface flags** | Information about the interface. Possible values are described in the "Interface Flags" section under *Common Output Fields Description*. | All levels |
| **Input Rate** | Input rate in bits per second (bps) and packets per second (pps). | None specified |
| **Output Rate** | Output rate in bps and pps. | None specified |
| **Statistics last cleared** | Time when the statistics for the interface were last set to zero. | **detail extensive** |
| **Traffic statistics** | Number and rate of bytes and packets received and transmitted on the physical interface.<br>• **Input bytes**—Number of bytes received on the interface.<br>• **Output bytes**—Number of bytes transmitted on the interface.<br>• **Input packets**—Number of packets received on the interface.<br>• **Output packets**—Number of packets transmitted on the interface. | **detail extensive** |

## Sample Output

### show interfaces (PIM De-Encapsulation)

```
user@host> show interfaces pd-0/0/0
Physical interface: pd-0/0/0, Enabled, Physical link is Up
  Interface index: 130, SNMP ifIndex: 25
  Type: PIMD, Link-level type: PIM-Decapsulator, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
```

### show interfaces brief (PIM De-Encapsulation)

```
user@host> show interfaces pd-0/0/0 brief
Physical interface: pd-0/0/0, Enabled, Physical link is Up
  Type: PIMD, Link-level type: PIM-Decapsulator, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: SNMP-Traps
```

### show interfaces detail (PIM De-Encapsulation)

```
user@host> show interfaces pd-0/0/0 detail
Physical interface: pd-0/0/0, Enabled, Physical link is Up
  Interface index: 130, SNMP ifIndex: 25, Generation: 11
  Type: PIMD, Link-level type: PIM-Decapsulator, MTU: Unlimited, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                   0                    0 bps
   Output bytes  :                   0                    0 bps
   Input  packets:                   0                    0 pps
   Output packets:                   0                    0 pps
```

### show interfaces extensive (PIM Encapsulation)

```
user@host> show interfaces pd-0/0/0 extensive
Physical interface: pd-0/0/0, Enabled, Physical link is Up
  Interface index: 130, SNMP ifIndex: 25, Generation: 11
  Type: PIMD, Link-level type: PIM-Decapsulator, MTU: Unlimited, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                   0                    0 bps
   Output bytes  :                   0                    0 bps
```

```
  Input  packets:                   0                    0 pps
  Output packets:                   0                    0 pps
```

### show interfaces (PIM Encapsulation)

```
user@host> show interfaces pe-0/0/0
Physical interface: pe-0/0/0, Enabled, Physical link is Up
  Interface index: 131, SNMP ifIndex: 26
  Type: PIME, Link-level type: PIM-Encapsulator, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
```

### show interfaces brief (PIM Encapsulation)

```
user@host> show interfaces pe-0/0/0 brief
Physical interface: pe-0/0/0, Enabled, Physical link is Up
  Type: PIME, Link-level type: PIM-Encapsulator, MTU: Unlimited, Speed: 800mbps
  Device flags   : Present Running
  Interface flags: SNMP-Traps
```

### show interfaces detail (PIM Encapsulation)

```
user@host> show interfaces pe-0/0/0 detail
Physical interface: pe-0/0/0, Enabled, Physical link is Up
  Interface index: 131, SNMP ifIndex: 26, Generation: 12
  Type: PIME, Link-level type: PIM-Encapsulator, MTU: Unlimited, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                  0                    0 bps
   Output bytes  :                  0                    0 bps
   Input  packets:                  0                    0 pps
   Output packets:                  0                    0 pps
```

### show interfaces extensive (PIM Encapsulation)

```
user@host> show interfaces pe-0/0/0 extensive
Physical interface: pe-0/0/0, Enabled, Physical link is Up
  Interface index: 131, SNMP ifIndex: 26, Generation: 12
  Type: PIME, Link-level type: PIM-Encapsulator, MTU: Unlimited, Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                   0                    0 bps
   Output bytes  :                   0                    0 bps
   Input  packets:                   0                    0 pps
   Output packets:                   0                    0 pps
```

## Release Information

Command introduced before Junos OS Release 7.4.

# show interfaces (Virtual Loopback Tunnel)

IN THIS SECTION

## Syntax

```
show interfaces vt-fpc/pic/port
<brief | detail | extensive | terse>
<descriptions>
<media>
<snmp-index snmp-index>
<statistics>
```

## Description

Display status information about the specified virtual loopback tunnel interface.

## Options

| | |
|---|---|
| **vt-*fpc*/*pic*/*port*** | Display standard information about the specified virtual loopback tunnel interface. |
| **brief \| detail \| extensive \| terse** | (Optional) Display the specified level of output. |
| **descriptions** | (Optional) Display interface description strings. |
| **media** | (Optional) Display media-specific information about network interfaces. |
| **snmp-index *snmp-index*** | (Optional) Display information for the specified SNMP index of the interface. |
| **statistics** | (Optional) Display static interface statistics. |

## Required Privilege Level

view

## Output Fields

Table 18 on page 352 lists the output fields for the `show interfaces` (virtual loopback tunnel) command. Output fields are listed in the approximate order in which they appear.

**Table 18: Virtual Loopback Tunnel show interfaces Output Fields**

| Field Name | Field Description | Level of Output |
| --- | --- | --- |
| Physical Interface | | |
| **Physical interface** | Name of the physical interface. | All levels |
| **Enabled** | State of the interface. Possible values are described in the “Enabled Field” section under *Common Output Fields Description*. | All levels |
| **Interface index** | Physical interface's index number, which reflects its initialization sequence. | **detail extensive** none |
| **SNMP ifIndex** | SNMP index number for the physical interface. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Type** | Type of interface. | All levels |
| **Link-level type** | Encapsulation used on the physical interface. | All levels |
| **MTU** | MTU size on the physical interface. | All levels |
| **Speed** | Speed at which the interface is running. | All levels |
| **Hold-times** | Current interface hold-time up and hold-time down, in milliseconds. | **detail extensive** |
| **Device flags** | Information about the physical device. Possible values are described in the “Device Flags” section under *Common Output Fields Description*. | All levels |

**Table 18: Virtual Loopback Tunnel show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Input Rate** | Input rate in bits per second (bps) and packets per second (pps). | None specified |
| **Output Rate** | Output rate in bps and pps. | None specified |
| **Statistics last cleared** | Time when the statistics for the interface were last set to zero. | **detail extensive** |
| **Traffic statistics** | Number and rate of bytes and packets received and transmitted on the physical interface.<br>• **Input bytes, Output bytes**—Number of bytes received and transmitted on the interface.<br>• **Input packets, Output packets**—Number of packets received and transmitted on the interface. | **detail extensive** |
| **Logical Interface** | | |
| **Logical interface** | Name of the logical interface. | All levels |
| **Index** | Logical interface index number, which reflects its initialization sequence. | **detail extensive** none |
| **SNMP ifIndex** | Logical interface SNMP interface index number. | **detail extensive** none |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Flags** | Information about the logical interface. Possible values are described in the “Interface Flags” section under *Common Output Fields Description*. | All levels |
| **Encapsulation** | Encapsulation on the logical interface. | All levels |
| **Input packets** | Number of packets received on the logical interface. | None specified |

Table 18: Virtual Loopback Tunnel show interfaces Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Output packets** | Number of packets transmitted on the logical interface. | None specified |
| **Bandwidth** | Bandwidth allotted to the logical interface, in kilobytes per second. | All levels |
| **Traffic statistics** | Total number of bytes and packets received and transmitted on the logical interface. These statistics are the sum of the local and transit statistics. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize.<br>• **Input bytes**—Number of bytes received on the interface.<br>• **Output bytes**—Number of bytes transmitted on the interface.<br>• **Input packets**—Number of packets received on the interface.<br>• **Output packets**—Number of packets transmitted on the interface. | **detail extensive** |
| **Transit statistics** | Statistics for traffic transiting the router. When a burst of traffic is received, the value in the output packet rate field might briefly exceed the peak cell rate. It takes awhile (generally, less than 1 second) for this counter to stabilize. | **detail extensive** |
| ***protocol-family*** | Protocol family configured on the logical interface. Possible values are described in the "Family Flags" section under *Common Output Fields Description*. | **brief** |
| **Protocol** | Protocol family configured on the logical interface. Possible values are described in the "Family Flags" section under *Common Output Fields Description*. | **detail extensive** none |
| **MTU** | Maximum transmission unit size on the logical interface. | **detail extensive** none |
| **Maximum labels** | Maximum number of MPLS labels configured for the MPLS protocol family on the logical interface. | **detail extensive** none |

Table 18: Virtual Loopback Tunnel show interfaces Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
| --- | --- | --- |
| **Generation** | Unique number for use by Juniper Networks technical support only. | **detail extensive** |
| **Route Table** | Routing table in which the logical interface address is located. For example, **0** refers to the routing table inet.0. | **detail extensive** |
| **Flags** | Information about protocol family flags. Possible values are described in the “Family Flags” section under *Common Output Fields Description*. | **detail extensive** none |

## Sample Output

### show interfaces (Virtual Loopback Tunnel)

```
user@host> show interfaces vt-1/2/0
Physical interface: vt-1/2/0, Enabled, Physical link is Up
  Interface index: 144, SNMP ifIndex: 40
  Type: Loopback, Link-level type: Virtual-loopback-tunnel, MTU: Unlimited,
  Speed: 800mbps
  Device flags   : Present Running
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)

  Logical interface vt-1/2/0.0 (Index 76) (SNMP ifIndex 57)
    Flags: Point-To-Point 16384 Encapsulation: Virtual-loopback-tunnel
  Input packets : 0
  Output packets: 0
    Protocol inet, MTU: Unlimited
      Flags: None
    Protocol mpls, MTU: Unlimited, Maximum labels: 3
      Flags: None
```

### show interfaces brief (Virtual Loopback Tunnel)

```
user@host> show interfaces vt-1/2/0 brief
Physical interface: vt-1/2/0, Enabled, Physical link is Up
  Type: Loopback, Link-level type: Virtual-loopback-tunnel, MTU: Unlimited,
  Speed: 800mbps
  Device flags   : Present Running

  Logical interface vt-1/2/0.0
    Flags: Point-To-Point 16384 Encapsulation: Virtual-loopback-tunnel
    inet
    mpls
```

### show interfaces detail (Virtual Loopback Tunnel)

```
user@host> show interfaces vt-1/2/0 detail
Physical interface: vt-1/2/0, Enabled, Physical link is Up
  Interface index: 144, SNMP ifIndex: 40, Generation: 27
  Type: Loopback, Link-level type: Virtual-loopback-tunnel, MTU: Unlimited,
  Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                  0                    0 bps
   Output bytes  :                  0                    0 bps
   Input  packets:                  0                    0 pps
   Output packets:                  0                    0 pps

  Logical interface vt-1/2/0.0 (Index 76) (SNMP ifIndex 57) (Generation 17)
    Flags: Point-To-Point 16384 Encapsulation: Virtual-loopback-tunnel
    Traffic statistics:
     Input  bytes  :                  0
     Output bytes  :                  0
     Input  packets:                  0
     Output packets:                  0
    Transit statistics:
     Input  bytes  :                  0                    0 bps
     Output bytes  :                  0                    0 bps
     Input  packets:                  0                    0 pps
```

```
    Output packets:                    0                    0 pps
   Protocol inet, MTU: Unlimited, Generation: 33, Route table: 0
     Flags: None
   Protocol mpls, MTU: Unlimited, Maximum labels: 3, Generation: 34, Route table: 0
     Flags: None
```

### show interfaces extensive (Virtual Loopback Tunnel)

```
user@host> show interfaces vt-1/2/0 extensive
Physical interface: vt-1/2/0, Enabled, Physical link is Up
  Interface index: 144, SNMP ifIndex: 40, Generation: 27
  Type: Loopback, Link-level type: Virtual-loopback-tunnel, MTU: Unlimited,
  Speed: 800mbps
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                   0                    0 bps
   Output bytes  :                   0                    0 bps
   Input  packets:                   0                    0 pps
   Output packets:                   0                    0 pps

  Logical interface vt-1/2/0.0 (Index 76) (SNMP ifIndex 57) (Generation 17)
    Flags: Point-To-Point 16384 Encapsulation: Virtual-loopback-tunnel
    Traffic statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Transit statistics:
     Input  bytes  :                   0                    0 bps
     Output bytes  :                   0                    0 bps
     Input  packets:                   0                    0 pps
     Output packets:                   0                    0 pps
    Protocol inet, MTU: Unlimited, Generation: 33, Route table: 0
      Flags: None
    Protocol mpls, MTU: Unlimited, Maximum labels: 3, Generation: 34, Route table: 0
      Flags: None
```

## Release Information

Command introduced before Junos OS Release 7.4.

# show interfaces fti

IN THIS SECTION



## Syntax

```
show interfaces fti number
<detail | extensive | terse | statistics >
```

## Description

Display status information about the specified FTIs.

## Options

| | |
|---|---|
| **fti *number*** | Display standard information about the specified FTIs. |
| **brief \| detail \| extensive \| terse** | (Optional) Display the specified level of output. |
| **descriptions** | (Optional) Display interface description strings. |
| **statistics** | (Optional) Display FTI statistics. |

## Required Privilege Level

view

## Output Fields

Table 19 on page 359 lists the output fields for the `show interfaces` (Flexible Tunnel Interfaces) command. Output fields are listed in the approximate order in which they appear.

**Table 19: FTI show interfaces Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Physical Interface** | | |
| `Physical interface` | Name of the physical interface and state of the interface. | All levels |
| `Enabled` | State of the physical interface. Possible values are described in the "Enabled Field" section under *Common Output Fields Description*. | All levels |
| `Interface index` | Index number of the physical interface, which reflects its initialization sequence. | All levels |
| `SNMP ifIndex` | SNMP index number for the physical interface. | All levels |

**Table 19: FTI show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| `Generation` | Unique number for use by Juniper Networks technical support only. | `detail extensive` |
| `Link-level type` | Encapsulation being used on the physical interface. Possible values: Flexible- tunnel-Interfaces | All levels |
| `MTU` | Maximum transmission unit size on the physical interface. | All levels |
| `Clocking` | Reference clock source of the interface. | All levels |
| `Speed` | Speed at which the interface is running. | All levels |
| `Device flags` | Information about the physical device. Possible values are described in the “Device Flags” section under *Common Output Fields Description*. | All levels |
| `Interface flags` | Information about the interface. Possible values are described in the “Interfaces Flags” section under *Common Output Fields Description*. | All levels |
| `Link-type` | Data transmission type. | All levels |
| `Link-flags` | Information about the link. Possible values are described in the “Link Flags” section under *Common Output Fields Description*. | All levels |
| `Last flapped` | Date, time, and how long ago the interface went from down to up or from up to down. The format is `Last flapped:` *`year-month-day hours:minutes:seconds timezone`* (*`hours:minutes:seconds`* `ago`). For example, `Last flapped: 2002-04-26 10:52:40 PDT (04:33:20 ago)`. | All levels |
| `Input packets` | Number of packets received on the physical interface. | All levels |
| `Output packets` | Number of packets transmitted on the physical interface. | All levels |
| `Physical-info` | Information about the physical interface. | `detail extensive` |

**Table 19: FTI show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| Hold-times | Current interface hold-time up and hold-time down. Value is in milliseconds. | detail extensive |
| Damping | Provides a smoothing of the up and down transitions (flapping) on an interface.<br>• halflife— Decay half-life. The number of seconds after which the accumulated interface penalty counter is reduced by half if the interface remains stable.<br>• max-suppress— Maximum hold-down time. The maximum number of seconds that an interface can be suppressed irrespective of how unstable the interface has been.<br>• reuse— Reuse threshold. When the accumulated interface penalty counter falls below this number, the interface is no longer suppressed.<br>• suppress— Cutoff (suppression) threshold. When the accumulated interface penalty counter exceeds this number, the interface is suppressed.<br>• state— Interface damping state. If damping is enabled on an interface, it is suppressed during interface flaps that match the configured damping parameters. | detail extensive |
| Current address | Configured MAC address. | detail extensive |
| Hardware address | Hardware MAC address. | detail extensive |
| Alternate-link-address | Backup link address. | detail extensive |
| Statistics last cleared | Time when the statistics for the interface were last set to zero. | detail extensive, statistics |

**Table 19: FTI show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| Traffic statistics | Number and rate of bytes and packets received and transmitted on the physical interface.<br>• Input bytes—Number of bytes and rate, in bps, at which bytes are received on the interface.<br>• Output bytes—Number of bytes and rate, in bps, at which bytes are transmitted on the interface.<br>• Input packets—Number of packets and rate, in pps, at which packets are received on the interface.<br>• Output packets—Number of packets and rate, in pps, at which packets are transmitted on the interface. | detail extensive |
| Input errors | Input errors on the interface:<br>• Errors—Sum of incoming frame terminates and frame checksum (FCS) errors.<br>• Drops—Number of packets dropped by the input queue of the I/O Manager ASIC. If the interface is saturated, this number increments once for every packet that is dropped by the ASIC's random early detection (RED) mechanism.<br>• Framing errors—Number of packets received with an invalid FCS.<br>• Runts—Number of frames received that are smaller than the runt threshold.<br>• Giants—Number of frames received that are larger than the giant threshold.<br>• Policed discards—Number of frames that the incoming packet match code discarded because they were not recognized or were not of interest. Usually, this field reports protocols that Junos OS does not handle.<br>• Resource errors—Sum of transmit drops. | detail extensive |

**Table 19: FTI show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
| --- | --- | --- |
| Output errors | Output errors on the interface. The following paragraphs explain the counters whose meaning might not be obvious:<br>• Carrier transitions —Number of times the interface has gone from down to up. This number does not normally increment quickly, increasing only when the cable is unplugged, the far-end system is powered down and then up, or another problem occurs. If the number of carrier transitions increments quickly (perhaps once every 10 seconds), then the cable, the far-end system, or the PIC is malfunctioning.<br>• Errors—Sum of the outgoing frame terminates and FCS errors.<br>• Drops—Number of packets dropped by the output queue of the I/O Manager ASIC. If the interface is saturated, this number increments once for every packet that is dropped by the ASIC's RED mechanism.<br>• MTU errors—Number of packets whose size exceeded the MTU of the interface.<br>• Resource errors—Sum of transmit drops. | detail extensive |
| IPv6 transit statistics | Number of IPv6 transit bytes and packets received and transmitted on the physical interface if IPv6 statistics tracking is enabled.<br>• Input bytes—Number of bytes received on the interface.<br>• Output bytes—Number of bytes transmitted on the interface.<br>• Input packets—Number of packets received on the interface.<br>• Output packets—Number of packets transmitted on the interface. | detail extensive |
| **Logical Interface** | | |
| Logical interface | Name of the logical interface. | All levels |
| Index | Index number of the logical interface (which reflects its initialization sequence). | detail extensive none |

**Table 19: FTI show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| `SNMP ifIndex` | SNMP interface index number of the logical interface. | `detail` `extensive` none |
| `Generation` | Unique number for use by Juniper Networks technical support only. | `detail` `extensive` |
| `Flags` | Information about the logical interface. Possible values are described in the "Logical Interface Flags Field" section under *Common Output Fields Description*. | All levels |
| `Encapsulation` | Encapsulation on the logical interface. | All levels |
| `Destination UDP Port` | Value to write to the destination-udp-port field. | All levels |
| `Source UDP Port Range` | Value to write to the source-udp-port-range field.<br>• min *min-port-number*: 1 through 65,535<br>• max *max-port-number*: 1 through 65,535 | All levels |
| `VNI` | Value to write to the `vni` attribute | All levels |
| `Source address` | IP address of the source side of the connection. | All levels |
| `Transit Statistics` | Statistics for traffic transiting the router.<br>• `Input bytes`—Number of bytes received on the interface.<br>• `Output bytes`—Number of bytes transmitted on the interface.<br>• `Input packets`—Number of packets received on the interface.<br>• `Output packets`—Number of packets transmitted on the interface. | `detail` `extensive` |
| `Protocol inet, MTU` | Protocol family configured on the logical interface. If the protocol is `inet`, the IP address of the interface is also displayed. MTU is the maximum transmission unit size on the logical interface | All levels |

**Table 19: FTI show interfaces Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| `Max nh cache` | Maximum interface neighbor discovery nexthop cache size. | All levels |
| `New hold nh limit` | Maximum number of new unresolved nexthops. | All levels |
| `Curr nh cnts` | Current number of resolved next-hop addresses. | All levels |
| `Curr new hold cnt` | Current number of unresolved next-hop addresses. | All levels |
| `NH drop cnt` | Number of neighbor discovery requests not serviced | All levels |
| `Flags` | Information about the logical interface. Possible values are described in the "Logical Interface Flags" section under *Common Output Fields Description*. | |
| `Route Table` | Routing table in which the logical interface address is located. For example, `0` refers to the routing table inet.0. | `detail extensive` |
| `Flags` | Information about protocol family flags. Possible values are described in the "Family Flags Field" section under *Common Output Fields Description*. | All levels |
| `Addresses, Flags` | Information about address flags. Possible values are described in the "Addresses Flags" section under *Common Output Fields Description*. | All levels |
| `Destination address` | IP address of the remote side of the connection. | All levels |
| `Local` | IP address of the logical interface. | All levels |
| `Broadcast` | Broadcast address of the logical interface. | `detail extensive` none |

## Sample Output

### show interfaces (Flexible Tunnel Interfaces on MX Series)

```
user@host> show interfaces fti0
Physical interface: fti0, Enabled, Physical link is Up
  Interface index: 136, SNMP ifIndex: 504
  Type: FTI, Link-level type: Flexible-tunnel-Interface, MTU: Unlimited, Speed: Unlimited
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Last flapped   : Never
    Input packets : 0
    Output packets: 0

  Logical interface fti0.0 (Index 340) (SNMP ifIndex 581)
    Flags: Up Point-To-Point SNMP-Traps Encapsulation: VXLAN-GPEv4
    Destination UDP port: 4789, VNI: 1000, Source address: 105.5.5.5, Destination address:
106.6.6.6
    Input packets : 0
    Output packets: 0
    Protocol inet, MTU: Unlimited
    Max nh cache: 0, New hold nh limit: 0, Curr nh cnt: 0, Curr new hold cnt: 0, NH drop cnt: 0
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 103.3.3/24, Local: 103.3.3.5, Broadcast: 103.3.3.255
```

### show interfaces (When source UDP port range for VXLAN encapsulation is configured at the Flexible Tunnel Interfaces on MX Series)

```
user@host> show interfaces fti0  Physical interface: fti0, Enabled, Physical link is Up
  Interface index: 137, SNMP ifIndex: 586
  Type: FTI, Link-level type: Flexible-tunnel-Interface, MTU: Unlimited, Speed: Unlimited
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Last flapped   : Never
    Input packets : 0
```

```
    Output packets: 0

  Logical interface fti0.0 (Index 329) (SNMP ifIndex 1678)
    Flags: Hardware-Down Up Point-To-Point SNMP-Traps Encapsulation: VXLAN-GPEv4
    Destination UDP port: 1000, Source UDP port range: [1000 - 2000],  VNI: 1000, Source
address: 13.13.13.13,     Destination address: 14.14.14.14
    Input packets : 0
    Output packets: 0
```

### show interfaces detail (Flexible Tunnel Interfaces on MX Series)

```
user@host> show interfaces fti0 detail
Physical interface: fti0, Enabled, Physical link is Up
  Interface index: 136, SNMP ifIndex: 504, Generation: 139
  Type: FTI, Link-level type: Flexible-tunnel-Interface, MTU: Unlimited, Clocking: Unspecified,
  Speed: Unlimited
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Physical info  : Unspecified
  Hold-times     : Up 0 ms, Down 0 ms
  Damping        : half-life: 0 sec, max-suppress: 0 sec, reuse: 0, suppress: 0, state:
unsuppressed
  Current address: Unspecified, Hardware address: Unspecified
  Alternate link address: Unspecified
  Last flapped   : Never
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                    0
   Output bytes  :                    0
   Input  packets:                    0
   Output packets:                    0
   IPv6 transit statistics:
   Input  bytes  :                    0
   Output bytes  :                    0
   Input  packets:                    0
   Output packets:                    0

  Logical interface fti0.0 (Index 340) (SNMP ifIndex 581) (Generation 149)
    Flags: Up Point-To-Point SNMP-Traps Encapsulation: VXLAN-GPEv4
```

```
    Destination UDP port: 4789, VNI: 1000, Source address: 10.5.5.5, Destination address:
10.6.6.6
    Traffic statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Local statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Transit statistics:
     Input  bytes  :                   0                    0 bps
     Output bytes  :                   0                    0 bps
     Input  packets:                   0                    0 pps
     Output packets:                   0                    0 pps
    Protocol inet, MTU: Unlimited
    Max nh cache: 0, New hold nh limit: 0, Curr nh cnt: 0, Curr new hold cnt: 0, NH drop cnt: 0
    Generation: 176, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.3.3/24, Local: 10.3.3.5, Broadcast: 10.3.3.255, Generation: 149

  Logical interface fti0.1 (Index 341) (SNMP ifIndex 582) (Generation 150)
    Flags: Up Point-To-Point SNMP-Traps Encapsulation: VXLAN-GPEv4
    Destination UDP port: 4789, VNI: 1000, Source address: 10.8.8.8, Destination address:
10.7.7.7
    Traffic statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Local statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Transit statistics:
     Input  bytes  :                   0                    0 bps
     Output bytes  :                   0                    0 bps
     Input  packets:                   0                    0 pps
     Output packets:                   0                    0 pps
```

```
    Protocol inet, MTU: Unlimited
    Max nh cache: 0, New hold nh limit: 0, Curr nh cnt: 0, Curr new hold cnt: 0, NH drop cnt: 0
    Generation: 177, Route table: 9
      Flags: Sendbcast-pkt-to-re, Is-Primary
      Addresses, Flags: Is-Default Is-Preferred Is-Primary
        Destination: 10.1.1/24, Local: 10.1.1.1, Broadcast: 10.1.1.255, Generation: 151
```

### show interfaces extensive (Flexible Tunnel Interfaces on MX Series)

```
user@host> show interfaces fti0 extensive
Physical interface: fti0, Enabled, Physical link is Up
  Interface index: 136, SNMP ifIndex: 504, Generation: 139
  Type: FTI, Link-level type: Flexible-tunnel-Interface, MTU: Unlimited, Clocking: Unspecified,
  Speed: Unlimited
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Physical info  : Unspecified
  Hold-times     : Up 0 ms, Down 0 ms
  Damping        : half-life: 0 sec, max-suppress: 0 sec, reuse: 0, suppress: 0, state:
unsuppressed
  Current address: Unspecified, Hardware address: Unspecified
  Alternate link address: Unspecified
  Last flapped   : Never
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                    0
   Output bytes  :                    0
   Input  packets:                    0
   Output packets:                    0
   IPv6 transit statistics:
   Input  bytes  :                    0
   Output bytes  :                    0
   Input  packets:                    0
   Output packets:                    0
  Input errors:
    Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0, Policed discards: 0,
    Resource errors: 0
  Output errors:
    Carrier transitions: 0, Errors: 0, Drops: 0, MTU errors: 0, Resource errors: 0
```

```
  Logical interface fti0.0 (Index 340) (SNMP ifIndex 581) (Generation 149)
    Flags: Up Point-To-Point SNMP-Traps Encapsulation: VXLAN-GPEv4
    Destination UDP port: 4789, VNI: 1000, Source address: 10.5.5.5, Destination address:
10.6.6.6
    Traffic statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Transit statistics:
     Input  bytes  :                    0                    0 bps
     Output bytes  :                    0                    0 bps
     Input  packets:                    0                    0 pps
     Output packets:                    0                    0 pps
    Protocol inet, MTU: Unlimited
    Max nh cache: 0, New hold nh limit: 0, Curr nh cnt: 0, Curr new hold cnt: 0, NH drop cnt: 0
    Generation: 176, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.3.3/24, Local: 10.3.3.5, Broadcast: 10.3.3.255, Generation: 149
```

### show interfaces terse (Flexible Tunnel Interfaces on MX Series)

```
user@host> show interfaces fti0 terse
Interface               Admin    Link      Proto    Local         Remote
        fti0                    up        up
        fti0.0                  up        up        inet     10.3.3.5/24
```

### show interfaces statistics (Flexible Tunnel Interfaces on MX Series)

```
user@host> show interfaces fti0 statistics
Physical interface: fti0, Enabled, Physical link is Up
  Interface index: 136, SNMP ifIndex: 504
```

```
  Type: FTI, Link-level type: Flexible-tunnel-Interface, MTU: Unlimited, Speed: Unlimited
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Last flapped   : Never
  Statistics last cleared: Never
    Input packets : 0
    Output packets: 0
  Input errors: 0, Output errors: 0

  Logical interface fti0.0 (Index 340) (SNMP ifIndex 581)
    Flags: Up Point-To-Point SNMP-Traps Encapsulation: VXLAN-GPEv4
    Destination UDP port: 4789, VNI: 1000, Source address: 10.5.5.5, Destination address:
10.6.6.6
    Input packets : 3
    Output packets: 3
    Protocol inet, MTU: Unlimited
    Max nh cache: 0, New hold nh limit: 0, Curr nh cnt: 0, Curr new hold cnt: 0, NH drop cnt: 0
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.3.3/24, Local: 10.3.3.5, Broadcast: 10.3.3.255
```

### show interfaces (Flexible Tunnel Interfaces on PTX and QFX Series)

```
user@host> show interfaces fti0 detail
Physical interface: ft0, Enabled, Physical link is Up
  Interface index: 193, SNMP ifIndex: 576, Generation: 196
  Type: FT, Link-level type: FT, MTU: Unlimited, Speed: Unlimited
  Hold-times     : Up 0 ms, Down 0 ms
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                   0                    0 bps
   Output bytes  :                   0                    0 bps
   Input  packets:                   0                    0 pps
   Output packets:                   0                    0 pps
   IPv6 transit statistics:
    Input  bytes  :                  0
```

```
    Output bytes  :                  0
    Input  packets:                  0
    Output packets:                  0
  Logical interface ft0.0 (Index 337) (SNMP ifIndex 583) (Generation 146)
    Flags: Hardware-Down Up Point-To-Point SNMP-Traps 0x4000
    Encapsulation: UDP
    Source: 10.1.1.1 Destination: 10.2.2.2 Destination-Port: 4789
    Traffic statistics:
     Input  bytes  :                  0
     Output bytes  :                  0
     Input  packets:                  0
     Output packets:                  0
    Local statistics:
     Input  bytes  :                  0
     Output bytes  :                  0
     Input  packets:                  0
     Output packets:                  0
    Transit statistics:
     Input  bytes  :                  0                    0 bps
     Output bytes  :                  0                    0 bps
     Input  packets:                  0                    0 pps
     Output packets:                  0                    0 pps
```

## Release Information

Command introduced in Junos OS Release 18.3R1.

### RELATED DOCUMENTATION

Flexible Tunnel Interfaces Overview | 38

Configuring Flexible Tunnel Interfaces | 45

Example: Configuring Flexible Tunnel Interfaces on MX Series Routers | 50

vni(Interfaces) | 252

# show ipsec certificates

IN THIS SECTION



## Syntax

```
show ipsec certificates
<brief | detail>
<crl crl-name | serial-number>
```

## Description

(Encryption interface on M Series and T Series routers only) Display information about the IPsec certificate database.

## Options

| | |
|---|---|
| none | Display standard information about all of the entries in the IPsec certificate database. |
| brief \| detail | (Optional) Display the specified level of output. |

| | |
|---|---|
| crl *crl-name* \| *serial-number* | (Optional) Display information about the entries on the certificate revocation list (CRL) or for the specified serial number. A CRL is a timestamped list identifying revoked certificates. The CRL is signed by a certificate authority (CA) or CRL issuer and made freely available in a public repository. Each revoked certificate is identified in a CRL by its certificate serial number. |

## Required Privilege Level

view

## Output Fields

Table 20 on page 374 lists the output fields for the `show ipsec certificates` command. Output fields are listed in the approximate order in which they appear.

**Table 20: show ipsec certificates Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Database** | Display information about the IPsec certificate database.<br>• **Total entries**—Number of database entries, including entries that are not trusted or that are in the process of being deleted.<br>• **Active entries**—Number of database entries, excluding entries that are marked as deleted.<br>• **Locked entries**—Number of statically configured database entries that cannot expire, such as CA certificates that are root or trusted. | All levels |
| **Subject** | Distinguished name for the certificate for **C**, **O**, **CN**, as described in RFC 3280, *Internet x.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.* | All levels |
| **ID** | Identification number of the database entry. **ID** is generated by the internal certificate database. | All levels |

**Table 20: show ipsec certificates Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **References** | Reference number the certificate manager has for the particular entry. | **detail** |
| **Serial** | Unique serial number assigned to each certificate by the CA. | All levels |
| **Flags** | State of the certificate.<br>• **Trusted**—Passed validity checks.<br>• **Not trusted**—Failed validity checks.<br>• **Root**—Entry is locked and may have been learned through IKE or a locally configured CA certificate.<br>• **Non-root**—Entry is not locked.<br>• **Crl-issuer**—Entity issues CRLs.<br>• **Non-crl-issuer**—Entity does not issue CRLs. | **detail** |
| **Validity period starts** | Start time that the certificate is valid, in the format *yyyy mon dd, hh:mm:ss* GMT. | **detail** |
| **Validity period ends** | End time that the certificate is valid, in the format *yyyy mon dd, hh:mm:ss* GMT. | **detail** |
| **Alternative name information** | Auxiliary identity for the certificate: *dns-name*, *email-address*, *ip-address*, or *uri* (uniform resource identifier). | **detail** |
| **Issuer** | Information about the entity that has signed and issued the CRL as described in RFC 2459, *Internet X.509 Public Key Infrastructure Certificate and CRL Profile.* | **detail** |

## Sample Output

### show ipsec certificates detail

```
user@host> show ipsec certificates detail
Database: Total entries: 3 Active entries: 4 Locked entries: 1
Subject: C=us, O=x
  ID: 5, References: 0, Serial: 22314868
  Flags: Trusted Non-root Crl-issuer
  Validity period starts: 2003 Mar  1st, 01:20:42 GMT
  Validity period ends: 2003 Mar 31st, 01:50:42 GMT
  Alternative name information:
    IP address: 10.20.210.1
  Issuer: C=FI, O=Company-ABC, CN=Company ABC class 2

Subject: C=us, O=x
  ID: 4, References: 0, Serial: 22315496
  Flags: Trusted Non-root Crl-issuer
  Validity period starts: 2003 Mar  1st, 01:21:45 GMT
  Validity period ends: 2003 Mar 31st, 01:51:45 GMT
  Alternative name information:
    IP address: 10.20.210.20
  Issuer: C=FI, O=Company-ABC, CN=Company ABC class 2

Subject: C=FI, O=SSH Company-ABC, CN=Company ABC class 2
  ID: 1, References: 1, Serial: 1538512
  Flags: Trusted Root Non-crl-issuer
  Validity period starts: 2001 Aug  1st, 07:08:32 GMT
  Validity period ends: 2004 Aug  1st, 07:08:32 GMT
  Alternative name information:
    Email address: certifier-support@ssh.com
  Issuer: C=FI, O=Company-ABC, CN=Company ABC class 2
```

## Release Information

Command introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

clear ipsec security-associations

# show ipsec redundancy

IN THIS SECTION



## Syntax

```
show ipsec redundancy (interface <es-fpc/pic/port> | security-associations <sa-name>)
```

## Description

(Encryption interface on M Series and T Series routers only) Display information about IPsec redundancy.

## Options

| | |
|---|---|
| interface <es-*fpc*/*pic*/*port*> | Display information about all encryption interfaces, or optionally, about a particular encryption interface. |
| security-associations <*sa-name*> | Display information about all remote tunnels, or optionally, about a particular remote tunnel. |

## Required Privilege Level

view

## Output Fields

Table 21 on page 378 lists the output fields for the `show ipsec redundancy` command. Output fields are listed in the approximate order in which they appear.

**Table 21: show ipsec redundancy Output Fields**

| Field Name | Field Description |
|---|---|
| **Failure counter** | Number of times a PIC switched between primary and backup interfaces, or the number of times the tunnel switched between the primary and remote peers since the software has been activated. |
| **Primary interface '** | Name of the interface configured to be the primary interface. |
| **Backup interface** | Name of the interface configured to be the backup interface. |
| **State** | State of the primary or backup interface can be **Active**, **Offline**, or **Standby**. Both ES PICs are initialized to **Offline**. For primary and remote peers, **State** can be **Active** or **Standby**. Both peers are in a state of **Standby** by default (there is not yet a connection between the two peers). |

**Table 21: show ipsec redundancy Output Fields *(Continued)***

| Field Name | Field Description |
|---|---|
| **Security association** | Name of the security association. |
| **Local IP** | Local IP address. |
| **Primary remote IP** | IP address of the configured primary remote peer. |
| **Backup remote IP** | IP address of the configured backup remote peer. |

## Sample Output

### show ipsec redundancy interface

```
user@host> show ipsec redundancy interface
    Failure counter: 0
    Primary interface: es-1/3/0, State: Active
    Backup interface : es-1/1/0, State: Standby
```

### show ipsec redundancy security-associations

```
user@host> show ipsec redundancy security-associations sa-dynamic
   Security association: sa-dynamic, Failure counter: 0
   Local IP: 192.0.2.4
   Primary remote IP: 198.51.100.5, State: Standby
   Backup remote IP : 192.0.2.3, State: Standby
```

## Release Information

Command introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

*request ipsec switch*

# show ipsec security-associations

IN THIS SECTION



## Syntax

```
show ipsec security-associations
<brief | detail>
<sa-name>
```

## Description

Display information about the IPsec security associations applied to the local or transit traffic stream.

## Options

| | |
|---|---|
| **none** | Display standard information about all IPsec security associations. |
| **brief \| detail** | (Optional) Display the specified level of output. |
| *sa-name* | (Optional) Display the specified IPsec security association. |

## Required Privilege Level

view

## Output Fields

Table 22 on page 381 lists the output fields for the **show ipsec security-associations** command. Output fields are listed in the approximate order in which they appear.

**Table 22: show ipsec security-associations Output Fields**

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Security association** | Name of the security association. | All levels |
| **Interface family** | Status of the interface family of the security association. If the interface family field is absent, it is a transport mode security association. The interface family can have one of three options:<br>• **Up**—The security association is referenced in the interface family and the interface family is up.<br>• **Down**—The security association is referenced in the interface family and the interface family is down.<br>• **No reference**—The security association is not referenced in the interface family. | All levels |

**Table 22: show ipsec security-associations Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Local gateway** | Gateway address of the local system. | All levels |
| **Remote gateway** | Gateway address of the remote system. | All levels |
| **Local identity** | Prefix and port number of the local end | All levels |
| **Remote identity** | Prefix and port number of the remote end. | All levels |
| **Direction** | Direction of the security association: **inbound** or **outbound**. | All levels |
| **SPI** | Value of the security parameter index. | All levels |
| **AUX-SPI** | Value of the auxiliary security parameter index.<br>• When the value is **AH** or **ESP**, **AUX-SPI** is always **0**.<br>• When the value is**AH+ESP**, **AUX-SPI** is always a positive integer. | All levels |
| **State** | Status of the security association:<br>• **Installed**—The security association is installed in the security association database. (For transport mode security associations, the value of **State** must always be **Installed**.)<br>• **Not installed**—The security association is not installed in the security association database. | **detail** |
| **Mode** | Mode of the security association:<br>• **transport**—Protects single host-to-host protections.<br>• **tunnel**—Protects connections between security gateways. | All levels |

Table 22: show ipsec security-associations Output Fields *(Continued)*

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Type** | Type of security association:.<br>• **manual**—Security parameters require no negotiation. They are static, and are configured by the user.<br>• **dynamic**—Security parameters are negotiated by the IKE protocol. Dynamic security associations are not supported in transport mode. | All levels |
| **Protocol** | Protocol supported:<br>• **transport** mode—Supports Encapsulation Security Protocol (**ESP**) or Authentication Header (**AH**).<br>• **tunnel** mode—Supports **ESP** or **AH+ESP**. | All levels |
| **Authentication** | Type of authentication used: **hmac-md5-96**, **hmac-sha1-96**, or **None**. | **detail** |
| **Encryption** | Type of encryption used: **des-cbc**, **3des-csc**, or **None**. | **detail** |
| **Soft lifetime**<br>**Hard lifetime** | (**dynamic** output only) Each lifetime of a security association has two display options, hard and soft, one of which must be present for a dynamic security association. The **hard lifetime** specifies the lifetime of the SA. The **soft lifetime**, which is derived from the hard lifetime, informs the IPsec key management system that the SA is about to expire. This allows the key management system to negotiate a new SA before the hard lifetime expires.<br>• **Expires in** *seconds* **seconds**—Number of seconds left until the security association expires.<br>• **Expires in** *kilobytes* **kilobytes**—Number of kilobytes left until the security association expires. | **detail** |
| **Anti-replay service** | State of the service that prevents packets from being replayed: **Enabled** or **Disabled**. | **detail** |

**Table 22: show ipsec security-associations Output Fields *(Continued)***

| Field Name | Field Description | Level of Output |
|---|---|---|
| **Replay window size** | Configured size, in packets, of the antireplay service window: **32** or **64**. The antireplay window size protects the receiver against replay attacks by rejecting old or duplicate packets. If the replay window size is **0**, the antireplay service is **disabled**. | **detail** |

## Sample Output

### show ipsec security-associations sa-name

```
user@host> show ipsec security-associations sa-cosmic brief
Security association: sa-cosmic, Interface family: Up
Local gateway: 192.0.2.1, Remote gateway: 198.51.100.1
Local identity: ipv4_subnet(any:0,[0..7]=0.0.0.0/0)
Remote identity: ipv4_subnet(any:0,[0..7]=0.0.0.0/0)
Direction SPI         AUX-SPI     Mode       Type     Protocol
inbound   2908734119  0           tunnel     dynamic  AH
outbound  3494029335  0           tunnel     dynamic  AH
```

### show ipsec security-associations sa-name detail

```
user@host> show ipsec security-associations sa-cosmic detail
Security association: sa-cosmic, Interface family: Up

Local gateway: 192.0.2.1, Remote gateway: 198.51.100.1
Local identity: ipv4_subnet(any:0,[0..7]=0.0.0.0/0)
Remote identity: ipv4_subnet(any:0,[0..7]=0.0.0.0/0)
Direction: inbound, SPI: 2908734119, AUX-SPI: 0, State: Installed
Mode: tunnel, Type: dynamic
Protocol: AH, Authentication: hmac-md5-96, Encryption: None
Soft lifetime: Expired
Hard lifetime: Expires in 120 seconds
Anti-replay service: Disabled
```

```
Direction: outbound, SPI: 3494029335, AUX-SPI: 0, State: Installed
Mode: tunnel, Type: dynamic
Protocol: AH, Authentication: hmac-md5-96, Encryption: None
Soft lifetime: Expired
Hard lifetime: Expires in 120 seconds
Anti-replay service: Disabled
```

## Release Information

Command introduced before Junos OS Release 7.4.

# show system certificate

IN THIS SECTION



## Syntax

```
show system certificate
<certificate-id>
```

## Description

(Encryption interface on M Series, T Series routers, QFX Series, and OCX Series switches only) Display installed certificates signed by the Juniper Networks certificate authority.

## Options

none — Display all installed certificates signed by the Juniper Networks certificate authority.

*certificate-id* — (Optional) Display the details of a particular certificate.

## Required Privilege Level

maintenance

## Output Fields

Table 23 on page 386 lists the output fields for the `show system certificate` command. Output fields are listed in the approximate order in which they appear.

**Table 23: show system certificate Output Fields**

| Field Name | Field Description |
|---|---|
| **Certificate identifier** | Unique identifier associated with a certificate. The certificate identifier is the common name of the subject. |

Table 23: show system certificate Output Fields *(Continued)*

| Field Name | Field Description |
| --- | --- |
| **Issuer**<br><br>**Subject** | Information about the certificate issuer and the distinguished name (**DN**) of the issuer, respectively:<br>• **Organization**—Name of the owner's organization.<br>• **Organizational unit**—Name of the owner's department.<br>• **Country**—Two-character country code in which the owner's system is located.<br>• **State**—State in the USA in which the owner is using the certificate.<br>• **Locality**—City in which the owner's system is located.<br>• **Common name**—Name of the owner of the certificate.<br>• **E-mail address**—E-mail address of the owner of the certificate. |
| **Validity** | When a certificate is valid. |
| **Signature algorithm** | Encryption algorithm applied to the installed certificate. |
| **Public key algorithm** | Encryption algorithm applied to the public key. |

## Sample Output

### show system certificate

```
user@host> show system certificate
Certificate identifier: Dallas-v3
  Issuer:
Organization: Juniper Networks, Organizational unit: Juniper CA,
Country:  US, State: CA, Locality: Sunnyvale, Common name: Dallas CA,
E-mail address:ca@example.com
  Subject:
```

```
  Organization: Juniper Networks, Organizational unit: Juniper CA,
Country: US, State: CA, Locality: Sunnyvale, Common name: Dallas-v3,
E-mail address:ca@example.com
Validity:
 Not before: Mar 13 03:23:25 2004 GMT
  Not after: Mar 24 03:23:25 2014 GMT
Signature algorithm: sha1WithRSAEncryption
Public key algorithm: dsaEncryption
```

### show system certificate (QFX Series)

```
user@host> show system certificate
Certificate identifier: Dallas-v3
  Issuer:
Organization: Juniper Networks, Organizational unit: Juniper CA,
Country:  US, State: CA, Locality: Sunnyvale, Common name: Dallas CA,
E-mail address:ca@example.com
  Subject:
  Organization: Juniper Networks, Organizational unit: Juniper CA,
Country: US, State: CA, Locality: Sunnyvale, Common name: Dallas-v3,
E-mail address:ca@example.com
Validity:
 Not before: Mar 13 03:23:25 2004 GMT
  Not after: Mar 24 03:23:25 2014 GMT
Signature algorithm: sha1WithRSAEncryption
Public key algorithm: dsaEncryption
```

## Release Information

Command introduced before Junos OS Release 7.4.